



CCSDS

The Consultative Committee for Space Data Systems

**Draft Recommendation for
Space Data System Standards**

**XML TELEMETRIC
AND COMMAND
EXCHANGE (XTCE)**

DRAFT RECOMMENDED STANDARD

CCSDS 660.0-R-2

**RED BOOK
December 2005**

AUTHORITY

| | |
|-----------|-------------------|
| Issue: | Red Book, Issue 2 |
| Date: | December 2005 |
| Location: | Not Applicable |

(WHEN THIS RECOMMENDED STANDARD IS FINALIZED, IT WILL CONTAIN THE FOLLOWING STATEMENT OF AUTHORITY:)

This document has been approved for publication by the Management Council of the Consultative Committee for Space Data Systems (CCSDS) and represents the consensus technical agreement of the participating CCSDS Member Agencies. The procedure for review and authorization of CCSDS documents is detailed in the *Procedures Manual for the Consultative Committee for Space Data Systems*, and the record of Agency participation in the authorization of this document can be obtained from the CCSDS Secretariat at the address below.

This document is published and maintained by:

CCSDS Secretariat
Office of Space Communication (Code M-3)
National Aeronautics and Space Administration
Washington, DC 20546, USA

STATEMENT OF INTENT

(WHEN THIS RECOMMENDED STANDARD IS FINALIZED, IT WILL CONTAIN THE FOLLOWING STATEMENT OF INTENT:)

The Consultative Committee for Space Data Systems (CCSDS) is an organization officially established by the management of its members. The Committee meets periodically to address data systems problems that are common to all participants, and to formulate sound technical solutions to these problems. Inasmuch as participation in the CCSDS is completely voluntary, the results of Committee actions are termed **Recommended Standards** and are not considered binding on any Agency.

This **Recommended Standard** is issued by, and represents the consensus of, the CCSDS members. Endorsement of this **Recommendation** is entirely voluntary. Endorsement, however, indicates the following understandings:

- o Whenever a member establishes a CCSDS-related **standard**, this **standard** will be in accord with the relevant **Recommended Standard**. Establishing such a **standard** does not preclude other provisions which a member may develop.
- o Whenever a member establishes a CCSDS-related **standard**, that member will provide other CCSDS members with the following information:
 - The **standard** itself.
 - The anticipated date of initial operational capability.
 - The anticipated duration of operational service.
- o Specific service arrangements shall be made via memoranda of agreement. Neither this **Recommended Standard** nor any ensuing **standard** is a substitute for a memorandum of agreement.

No later than five years from its date of issuance, this **Recommended Standard** will be reviewed by the CCSDS to determine whether it should: (1) remain in effect without change; (2) be changed to reflect the impact of new technologies, new requirements, or new directions; or (3) be retired or canceled.

In those instances when a new version of a **Recommended Standard** is issued, existing CCSDS-related member standards and implementations are not negated or deemed to be non-CCSDS compatible. It is the responsibility of each member to determine when such standards or implementations are to be modified. Each member is, however, strongly encouraged to direct planning for its new standards and implementations towards the later version of the Recommended Standard.

FOREWORD

Through the process of normal evolution, it is expected that expansion, deletion, or modification of this document may occur. This Recommended Standard is therefore subject to CCSDS document management and change control procedures, which are defined in the *Procedures Manual for the Consultative Committee for Space Data Systems*. Current versions of CCSDS documents are maintained at the CCSDS Web site:

<http://www.ccsds.org/>

Questions relating to the contents or status of this document should be addressed to the CCSDS Secretariat at the address indicated on page i.

At time of publication, the active Member and Observer Agencies of the CCSDS were:

Member Agencies

- Agenzia Spaziale Italiana (ASI)/Italy.
- British National Space Centre (BNSC)/United Kingdom.
- Canadian Space Agency (CSA)/Canada.
- Centre National d'Etudes Spatiales (CNES)/France.
- Deutsches Zentrum für Luft- und Raumfahrt e.V. (DLR)/Germany.
- European Space Agency (ESA)/Europe.
- Federal Space Agency (Roskosmos)/Russian Federation.
- Instituto Nacional de Pesquisas Espaciais (INPE)/Brazil.
- Japan Aerospace Exploration Agency (JAXA)/Japan.
- National Aeronautics and Space Administration (NASA)/USA.

Observer Agencies

- Austrian Space Agency (ASA)/Austria.
- Belgian Federal Science Policy Office (BFSP0)/Belgium.
- Central Research Institute of Machine Building (TsNIIMash)/Russian Federation.
- Centro Tecnico Aeroespacial (CTA)/Brazil.
- Chinese Academy of Space Technology (CAST)/China.
- Commonwealth Scientific and Industrial Research Organization (CSIRO)/Australia.
- Danish Space Research Institute (DSRI)/Denmark.
- European Organization for the Exploitation of Meteorological Satellites (EUMETSAT)/Europe.
- European Telecommunications Satellite Organization (EUTELSAT)/Europe.
- Hellenic National Space Committee (HNSC)/Greece.
- Indian Space Research Organization (ISRO)/India.
- Institute of Space Research (IKI)/Russian Federation.
- KFKI Research Institute for Particle & Nuclear Physics (KFKI)/Hungary.
- Korea Aerospace Research Institute (KARI)/Korea.
- MIKOMTEK: CSIR (CSIR)/Republic of South Africa.
- Ministry of Communications (MOC)/Israel.
- National Institute of Information and Communications Technology (NICT)/Japan.
- National Oceanic & Atmospheric Administration (NOAA)/USA.
- National Space Program Office (NSPO)/Taipei.
- Space and Upper Atmosphere Research Commission (SUPARCO)/Pakistan.
- Swedish Space Corporation (SSC)/Sweden.
- United States Geological Survey (USGS)/USA.

PREFACE

This document is a draft CCSDS Recommended Standard. Its 'Red Book' status indicates that the CCSDS believes the document to be technically mature and has released it for formal review by appropriate technical organizations. As such, its technical contents are not stable, and several iterations of it may occur in response to comments received during the review process.

Implementers are cautioned **not** to fabricate any final equipment in accordance with this document's technical content.

DOCUMENT CONTROL

| Document | Title | Date | Status |
|--------------------|---|------------------|---------------|
| CCSDS 660.0-R-2 | XML Telemetric and Command Exchange (XTCE), Draft Recommended Standard, Issue 2 | December 2005 | Current draft |

CONTENTS

| <u>Section</u> | <u>Page</u> |
|---|-------------|
| XML TELEMETRIC AND COMMAND EXCHANGE (XTCE) | 1 |

XML Telemetric and Command Exchange (XTCE)

Version 1.1 (draft 1)

Contents

| | |
|--|----|
| XML Telemetric and Command Exchange (XTCE)..... | 1 |
| Contents..... | 2 |
| Foreword | 5 |
| Introduction | 6 |
| 1 Scope..... | 8 |
| 2 Conformance | 8 |
| 3 Normative references..... | 8 |
| 4 Terms and definitions | 9 |
| 5 Symbols (and abbreviated terms) | 9 |
| 6 The Specification..... | 10 |
| 6.1 The Root Object – The SpaceSystem | 10 |
| 6.1.1 The Header Record..... | 11 |
| 6.1.2 TelemetryMetaData | 11 |
| 6.1.2.1 ParameterTypeSet..... | 13 |
| 6.1.2.2 ParameterSet..... | 13 |
| 6.1.2.3 ContainerSet..... | 14 |
| 6.1.2.4 MessageSet | 14 |
| 6.1.2.5 StreamSet..... | 14 |
| 6.1.3.6 AlgorithmSet..... | 16 |
| 6.1.3 CommandMetaData..... | 17 |
| 6.1.3.1 ArgumentTypeSet..... | 17 |
| 6.1.3.2 MetaCommandSet | 18 |
| 6.1.3.2.1 BaseMetaCommand | 19 |
| 6.1.3.2.2 ArgumentList | 19 |
| 6.1.3.2.3 CommandContainer..... | 19 |
| 6.1.3.2.4 TransmissionConstraintList..... | 19 |
| 6.1.3.2.5 DefaultSignificance and ContextSignificanceList..... | 19 |
| 6.1.3.2.6 Interlock | 19 |
| 6.1.3.2.6 Verifiers..... | 19 |
| 6.1.3.2.7 ParameterToSetList | 19 |
| 6.1.4 ServiceSet | 19 |
| 6.1.6 Defaults..... | 20 |
| 6.2 The Schema | 20 |
| Annex A -The SpaceSystem Schema | 21 |

DRAFT RECOMMENDED STANDARD FOR XML TELEMETRIC AND COMMAND EXCHANGE

Annex B - Schema Style Notes 74
Bibliography 75

DRAFT RECOMMENDED STANDARD FOR XML TELEMETRIC AND COMMAND EXCHANGE

Lockheed Martin, The Boeing Company, and The European Space Agency waive copyright on this document to the Object Management Group (OMG) and the OMG members for unlimited duplication.

For enquiries please contact:

Gerry Simon
Lockheed Martin – Mission Systems
Center For Research Support (CERES)
720 Irwin Avenue
Schriever AFB, CO 80912
USA
719-567-8349
gerry.simon@schriever.af.mil

Janice Ann Champion, Senior Staff Engineer
Boeing Satellite Systems
W/EO1/D110
P.O. Box 92919
Los Angeles, CA 90009
(310) 416-4544
janice.a.champion@boeing.com

Mario Merri,
European Space Operations Centre
Robert Bosch Strasse 5,
64293 Darmstadt, Germany
Tel: +49 6151 90 2292
Fax: +49 6151 90 3010
[Email: Mario.Merri@esa.int](mailto:Mario.Merri@esa.int)

Foreword

This XML Telemetric and Command Exchange (XTCE) data specification answers the need for an information model and data exchange format for telemetry and commanding in all phases of the spacecraft, payload, and ground segment lifecycle: system design, development, test, validation, and mission operations.

This specification addresses a compelling need for a standard exchange format recognized independently by each of its authors and contributors. Lockheed Martin, ESA, Boeing, NASA GSFC, USAF SMC, Harris, Raytheon, SciSys, CSC and GST have all made significant contributions representing a wide and varied sampling of the space industry.

Space mission implementations face a very dynamic environment with fast-paced information technology advancement and shrinking space budgets. A more focused use of decreasing public investments in space requires a cost reduction over their entire lifecycle, from development up to the end of the useful life of a spacecraft. The use of standards specifications from the early stages of satellite development through mission operation will reduce life-cycle cost.

The XTCE specification is intended to provide a robust, international standard for data exchange, one that can be easily become a central element in a simplified contract to Space System providers for Telemetry and Command definition – from simple space components to entire constellations.

Satellite design and development is performed today through the use of a number of disparate tools and techniques. Interface design to satellite systems and to the payloads the satellites are housing is still a manual and time-consuming effort. Data design, both telemetry and commanding, is still performed multiple times by multiple contractors during the lifecycle of the satellite, well before the satellite is ever deployed for mission operations. The standardization of satellite telemetry and command data for spacecraft health and safety, as well as payload interfaces will reduce the cost of these implementations as well as decrease the schedule of development, integration, and test of the satellite and its component systems. This specification can also be used to support multiple, heterogeneous missions, facilitating interoperability between ground control systems, simulators, testing facilities, etc.

At the heart of this specification is a very robust information model for telemetry and commanding that will support all phases of the satellite, payload, and ground segment lifecycle: system design, development, test, validation, and mission operations.

Introduction

Purpose: This specification is an information model for spacecraft telemetry and commanding data. For a given mission there are a number of lifecycle phases that are supported by a variety of systems and organizations. Additionally, many of these organizations support multiple heterogeneous missions using a common ground segment infrastructure. Telemetry and command definitions must be exchanged among all of these phases, systems, and organizations. This is made difficult and costly because there is no standard format for exchanging this information. The lack of standardization currently requires custom ingestion of the telemetry and commanding information. This customization is inherently error-prone, resulting in the need to revalidate at each step in the lifecycle.

A typical example of this process is between the spacecraft manufacturer and spacecraft-operating agency. The spacecraft manufacturer defines the telemetry and command data in a format that is much different than the one used in the ground segment. This creates the need for database translation, increased testing, software customization, and increased probability of error. Standardization of the command and telemetry data definition format will streamline the process allowing dissimilar systems to communicate without the need for the development of mission specific database import/export tools.

Ideally, a spacecraft operator should be able to transition a spacecraft mission from one ground system to another by simply moving an already existing command and telemetry database compliant with this command and telemetry database specification to another ground system which equally supports this command and telemetry database specification..

In addition, standardization will enable space or ground segment simulators to more easily support multiple heterogeneous missions.

XTCE provides a standard format for defining the Telemetric and Telecommand (TM/TC) data required to perform the processing shown in figure 1.

Overview: The normative portion of this specification is presented as a single XML schema compliant with the W3C recommendation of 02/05/2001. The schema is found in Annex A or may be obtained as an independent convenience document.

The schema has an object-oriented structure where all the elements of this specification belong to a single root object – the SpaceSystem.

Philosophy: The space industry is currently divided between Packet telemetry and commanding and Time Division Multiplexing (TDM) telemetry and commanding. While the basic construction of either TDM or packet telemetry is fundamentally not all that different, nomenclature differences between the two give the appearance of a larger divide. The XTCE specification avoids using nomenclature from either the TDM or packet worlds to avoid any possible confusion; terms like ‘minor frame’, ‘major frame’, or ‘packet’ are nowhere in this specification other than in examples. Furthermore, the XTCE specification does not itself use any existing packet or TDM standards (such as CCSDS packet formats, or IRIG-106 minor frame standards), but it does provide a mechanism to use XTCE to build libraries of available containers that represent these standards.

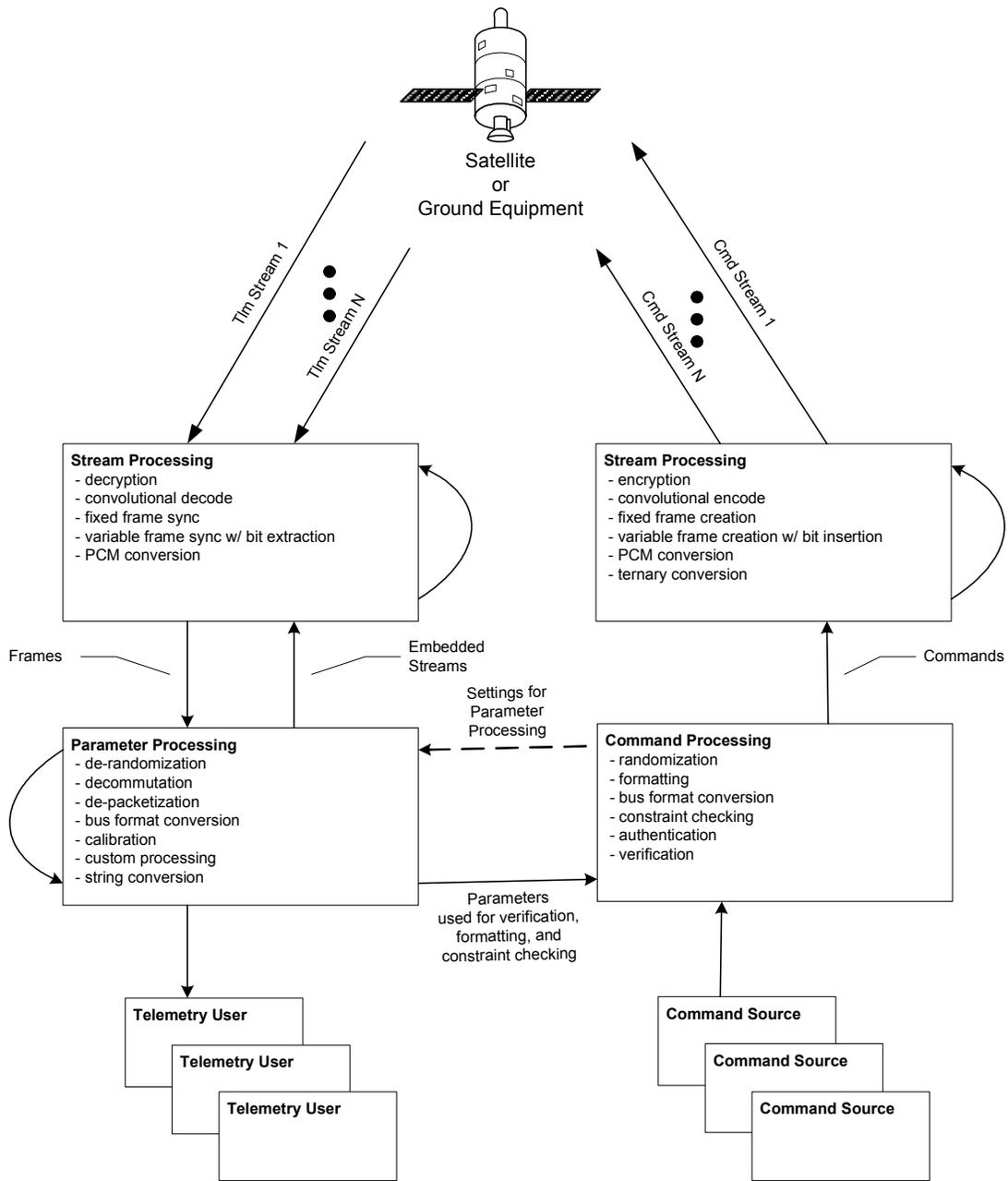


Figure 1 - Telemetric and Command Processing Meta Data Encapsulated in XTCE XML

1 Scope

This specification addresses the need for a standardized information model capable of supporting TM/TC definitions across the widest possible range of space domain activities. The goal is to allow TM/TC definitions to be exchanged between different organizations and systems, often at the boundaries of mission phases, without the need for customized import/export, re-validation, or even re-implementation of mission databases.

The scope of this specification is limited to satellite telemetry and commanding data constructs necessary to support satellite and payload data design:

- Telemetry data definitions including support for CCSDS packets as well as TDM frames.
- Data manipulation algorithms to support packaging and unpacking of individual data items.
- Commanding data definitions including command identification, argument specification, and validation criteria.
- Data representation definitions
- Data properties including such things as its default value, validity criteria, and data dependencies.
- The definition of extensible formats such that blocks of information (whether frames of data that are not decommutated or object references or object method calls) can be portrayed in this architecture.

The scope of this specification does not extend to:

- Data distribution mechanisms.
- Command and data protocol specifications.
- RF or analog stream characterization
- Data grouping including aggregation and coherent data sets
- Data representation (visualization properties)
- Scheduling configuration properties
- Orbital properties

This specification addresses only the definition of TM/TC data, and not the transfer of live or historical TM/TC data.

2 Conformance

The schema (.xsd file) in Annex A is normative. A compliant database is an XML file that complies with this schema. Fully compliant implementing software will interpret and/or generate any databases compliant with this specification. Compliant implementing software will interpret and/or generate all database elements required by the schema.

3 Normative references

The following normative documents contain provisions which, through reference in this text, constitute provisions of this specification. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply.

| | |
|---|--|
| http://www.w3.org/TR/REC-xml | W3C Recommendation - Extensible Markup Language (XML) 1.0 (Second Edition, 6 October 2000) |
|---|--|

| | |
|---|---|
| http://www.w3.org/TR/xmlschema-0/ | W3C Recommendation - XML Schema Part 0: Primer (2 May 2001) |
| http://www.w3.org/TR/xmlschema-1/ | W3C Recommendation - XML Schema Part 1: Structures (2 May 2001) |
| http://www.w3.org/TR/xmlschema-2/ | W3C Recommendation - XML Schema Part 2: Datatypes (2 May 2001) |

4 Terms and definitions

For the purposes of this specification, the following terms and definitions apply.

Telemetry

(from IEEE Std 1000 [1972]) “Measurement with the aid of intermediate means that permit the measurement to be interpreted at a distance from the primary detector.” All measurements on board the spacecraft are transmitted to the ground system in a telemetry stream. Telemetry as used here refers to these measurements whether on-board the spacecraft or transmitted to the ground system. Most telemetry measurements will require engineering unit conversion and measurements will have associated validation ranges or lists of acceptable values.

Commands

Messages that instruct an action on a remote system. Spacecraft commanding usually implies coding and packaging of the command information, validation and verification, as well as authorization to perform. Telemetry and Commanding data are necessarily related to one another, with some command information originating from telemetry and commands relating to particular telemetry measurements. Therefore, the ability to relate individual telemetry with one another and to commands is a very important part of this specification. Packaging of both telemetry and commands can be performed in a number of ways. The most common way to package data for transmission is to use the CCSDS Telemetry and Commanding Packaging format.

5 Symbols (and abbreviated terms)

List of symbols/abbreviations

In general, the XTCE specification favors expressive, fully spelled out terms over abbreviated notation. The exceptions are modifiers used as prefixes or postfixes to objects used within the schema, and of course ‘XTCE’ the name of the standard itself. These terms are listed below.

Abbreviations:

Parm – is an abbreviation sometimes used for Parameter.

XTCE – XML Telemetric and Command Exchange format.

Prefixes and Postfixes

Meta – Is a description. For example a MetaCommand is a command description.

Set – Is an unordered collection. For example a MetaCommandSet is an unordered collection of command descriptions.

List – Is an ordered collection. For example an ArgumentList is an ordered collection of arguments.

Ref – is a reference (by name) to an object defined elsewhere in the XML document. For example an **ArgumentRef** is a named reference to an **Argument** defined elsewhere.

6 The Specification

6.1 The Root Object – The SpaceSystem

Recognizing that spacecraft operations involve much more than simply controlling the spacecraft, the top-level object is not ‘Spacecraft’ but the more generic term ‘SpaceSystem’. This name provides deference to the fact that a spacecraft operations center must control antennas, recorders, ground processing equipment, RF hardware and many other assets that may use this data specification; each of these objects is a ‘SpaceSystem’. A SpaceSystem, like all of the major objects in an XTCE database, may have a short description, a long description (that may contain HTML markup documentation), and a list of alias names. A SpaceSystem may have a Header, zero or more sub-SpaceSystems, CommandMetaData and TelemetryMetaData. The CommandMetaData and TelemetryMetaData components contain the bulk of the Telemetric and Command data. The sub-SpaceSystems give the data a hierarchical structure.

Note on the sub-SpaceSystem and the hierarchical structure

Because a SpaceSystem may itself contain other SpaceSystems, the data may also have a hierarchical structure – similar to the structure of a real space system. The hierarchical organization offers several important advantages over a flat entity list:

- Fewer name space collisions – Almost every spacecraft contains redundant components for reliability or to accomplish the mission. A communications spacecraft may have a dozen transponders each with the same set of telemetry points and commands. In a flat namespace each of those telemetry points needs to be mapped into a unique name. Using a hierarchical namespace, those identical telemetry points can be simply placed into separate sub-SpaceSystems.
- Better organization – modern spacecraft typically have thousands of commands and tens of thousands of telemetry parameters; this number is trending upward. The directory structure provided by this specification provides an improved way to manage this large volume of data. Each subsystem developer can deliver SpaceSystems representing their subsystem without integration issues.
- Defaults at the SpaceSystem level – many of the attributes needed to define spacecraft parameters (e.g. bit order, byte order) are common to most of the parameters in the spacecraft or spacecraft sub-system. This specification allows these attributes to be assigned at the directory level, thereby avoiding their repetition in each parameter.
- Spacecraft which are normally thought of as a SpaceSystem, may actually be sub-SpaceSystems for a constellation of spacecraft SpaceSystems.
- Natural hierarchy – spacecraft designs are increasing in complexity and are normally comprised of systems of systems. The hierarchical organization allowed by a directory structure reflects this.

Note on Names

Parameter, and MetaCommand and other major entity names within this database may be any length but are prohibited from containing the ‘/’, ‘.’, and ‘:’ characters as these are reserved. The ‘/’ is used as the SpaceSystem separator (Unix and HTTP style). The ‘.’ is reserved for future use as a selector for data from other SpaceSystems. The ‘:’ is reserved as an attribute selector.

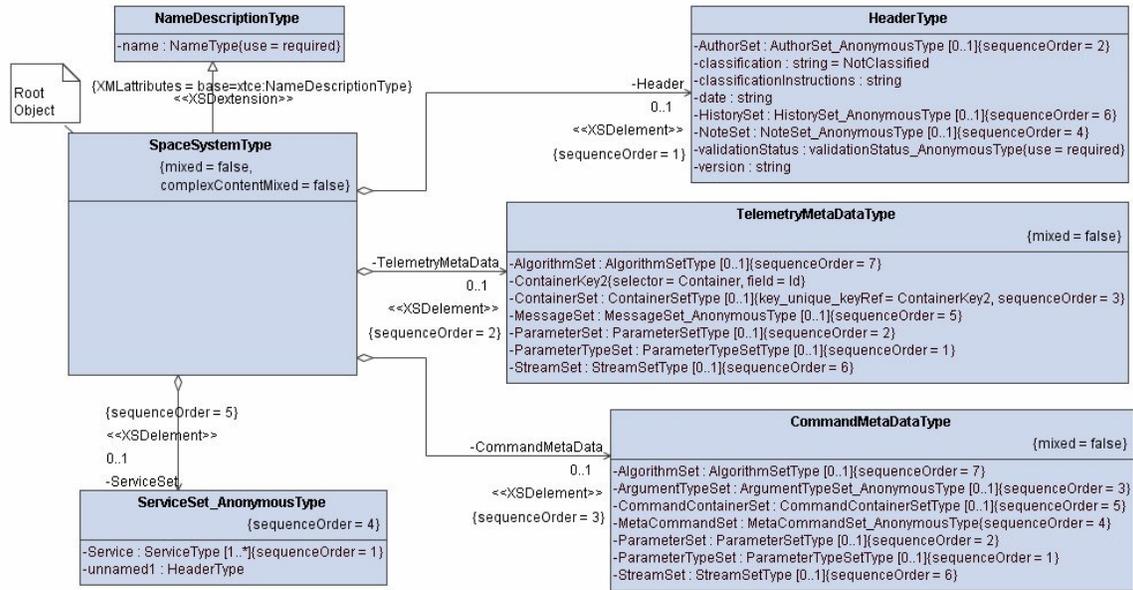


Figure 2 SpaceSystem

6.1.1 The Header Record

A SpaceSystem, may contain an optional header record. This record contains some basic context on the data itself (e.g. source, version, revision history, notes, and classification).

6.1.2 TelemetryMetaData

Because Telemetry and Command databases are frequently developed and maintained independently, the XTCE format divides TelemetryMetaData and CommandMetaData into separate, but similar sections. TelemetryMetaData is really nothing more than a grouping for data about Telemetry. TelemetryMetaData has a ParameterTypeSet, a ParameterSet, a ContainerSet, a MessageSet, a StreamSet, and an AlgorithmSet. Following are descriptions of these collection types.

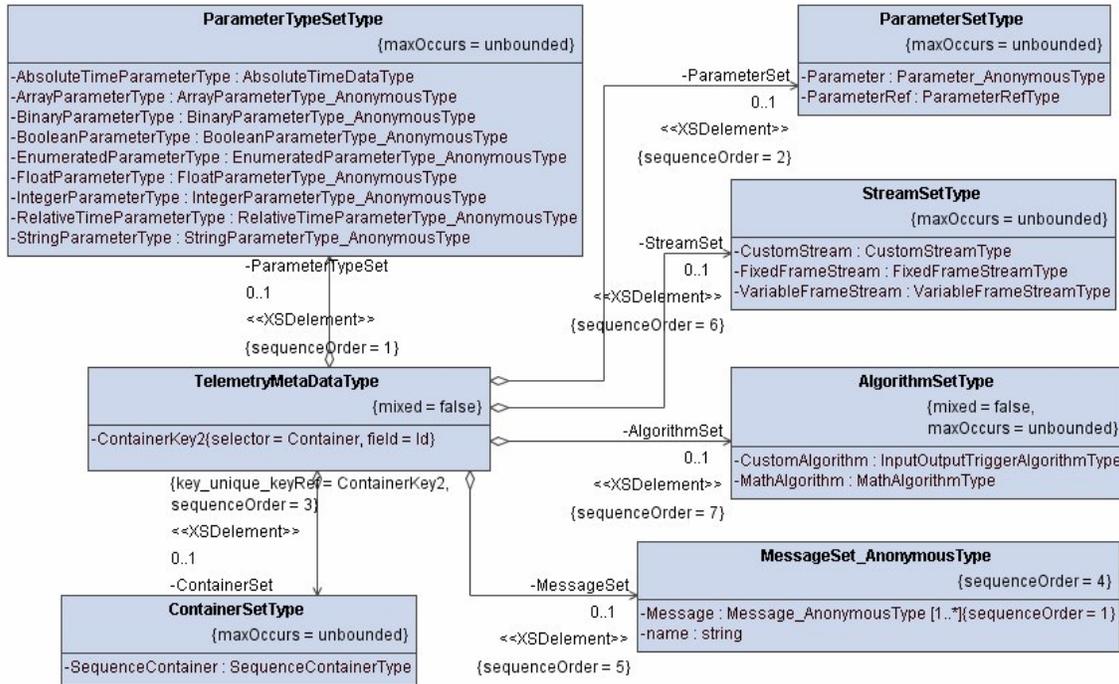


Figure 3 Telemetry MetaData

6.1.2.1 ParameterTypeSet

A ParameterTypeSet is an unordered collection of ParameterTypes. ParameterTypes are the MetaData for Parameters; ParameterTypes are instantiated to create Parameters. ParameterType is the description of something that can have a value (a Parameter). Information contained in ParameterType includes the data type, description, alarm limits, engineering units and string conversion specifications and calibrations. Parameters may be or variable length. Most Parameters are telemetered parameters (a.k.a measurands) and must also include information about how the Parameter value is encoded for transmission. This information includes size in bits, byte order, data type, and parity checks. All of the encoding information is in one of four different 'DataEncoding' elements. XTCE supports four different types of Encoding: IntegerDataEncoding, FloatDataEncoding, StringEncoding and BinaryDataEncoding. Note that the data encoding type only speaks to how the Parameter (or Command argument) is transmitted, not how it is handled on the SpaceSystem or ground.

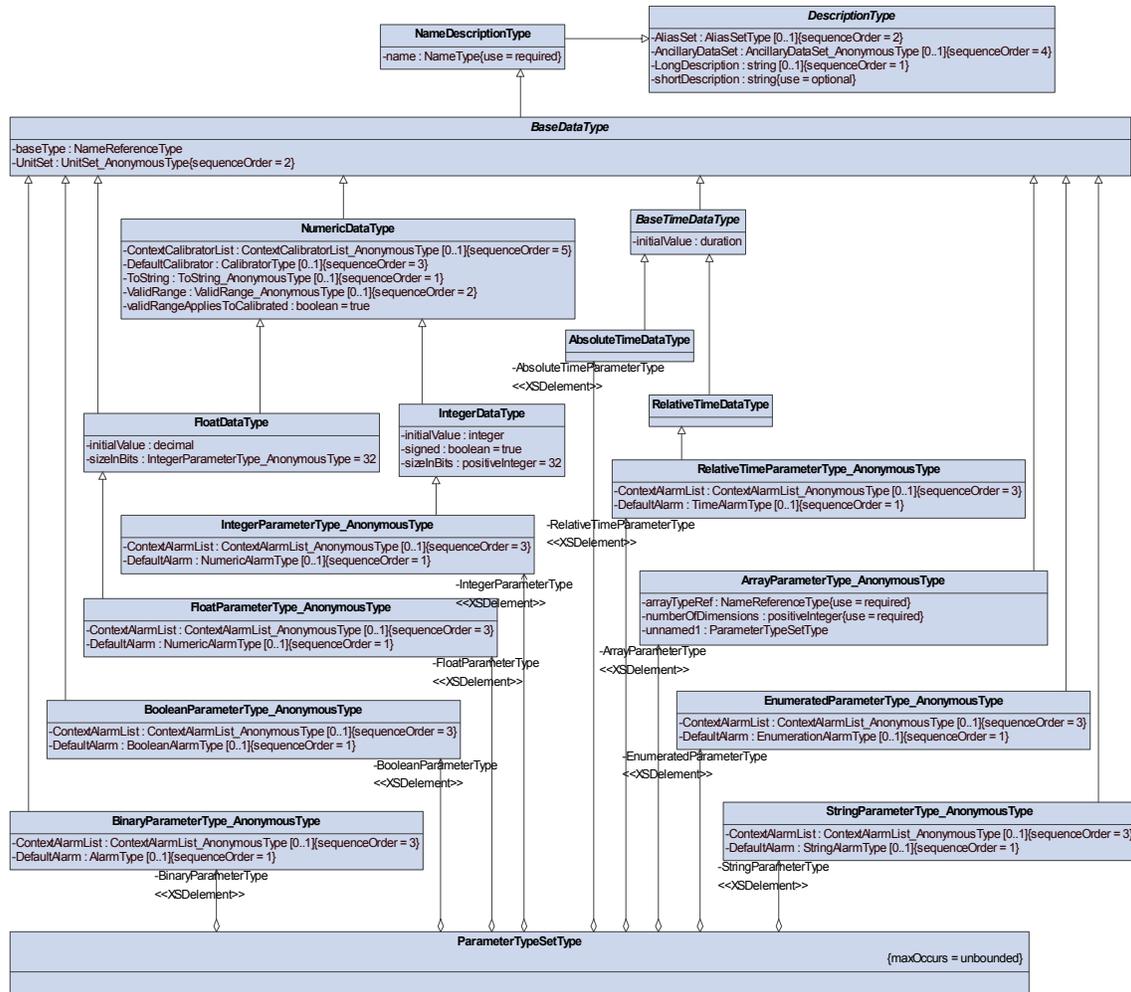


Figure 4 ParameterTypeSet

6.1.2.2 ParameterSet

A ParameterSet is an unordered collection of Parameters. Parameters are instantiations of ParameterTypes. Parameters are normally a very simple name and reference to a ParameterType. Parameters may also have alias names and may have properties unique to that instantiation. At any point in time (instance) a Parameter has a value; a Parameter is not the value itself. Parameter names are case sensitive, may be any

length, cannot contain spaces ‘ ’, ‘/’, ‘[’ or ‘]’ as these are reserved characters. The aliases have no restrictions.

6.1.2.3 ContainerSet

A ContainerSet is an unordered collection of SequenceContainers. SequenceContainers are a simple, yet very powerful means define TDM telemetry streams (to any commutation level), packetized streams or many other package formats. A SequenceContainer contains (in the EntryList) an ordered list of raw parameters, parameter segments, stream segments, containers, or container segments. SequenceContainers may inherit from other sequence containers. The inheritance aspect of SequenceContainers is useful not only for minimizing the effort required to describe a family of SequenceContainers, but is also a very powerful and expressive means of container identification – the process of distinguishing one container from others (e.g. minorFrame 20 is a type of minorFrame where the minor frame counter equals 20). SequenceContainer inheritance may be arbitrarily deep.

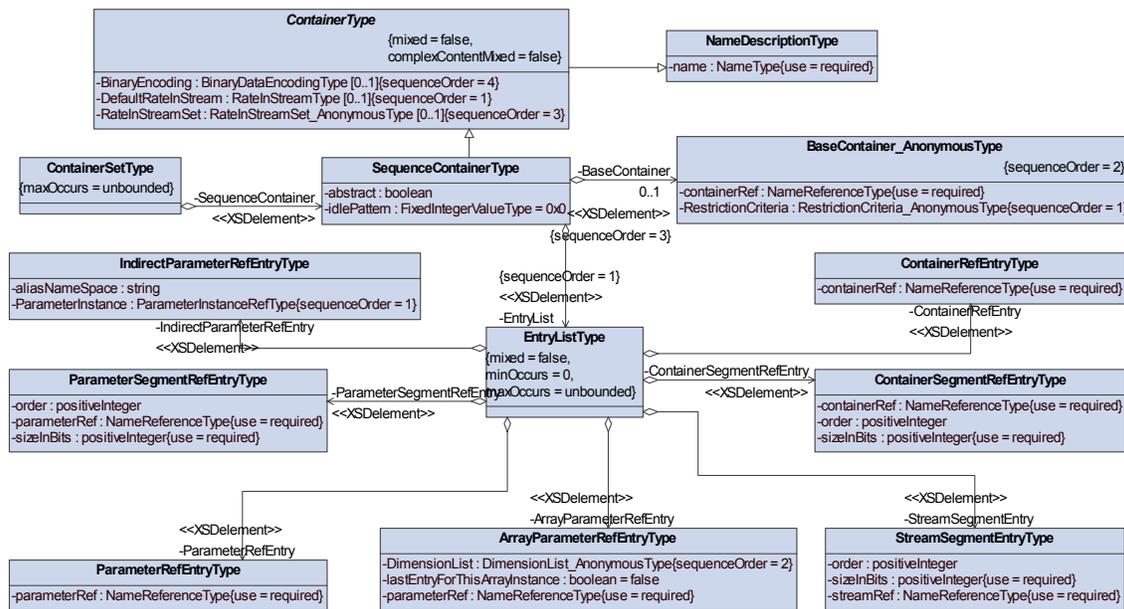


Figure 5 ContainerSet

A SequenceContainer may represent a packet, a frame, a sub-frame or any other grouping/structure of data items. The simple form of a Sequence element is an ordered set of Parameter References or other Container References.

6.1.2.4 MessageSet

A MessageSet is an unordered collection of Messages. Messages are an alternative method of uniquely identifying containers within a Service. A message provides a test in the form of MatchCriteria to match to a container. A simple example might be: When minorframeID=21, the message is the 21st minorframe container. The collection of messages to search through will be bound by a Service.

6.1.2.5 StreamSet

A StreamSet is an unordered collection of Streams. Spacecraft uplinks and spacecraft downlinks are digital streams of data and there are a number of processing functions that are done on the stream level. The StreamSet in a SpaceSystem XTCE document can contain all of the information on how to assemble, disassemble and process spacecraft uplink and downlink streams for that SpaceSystem.

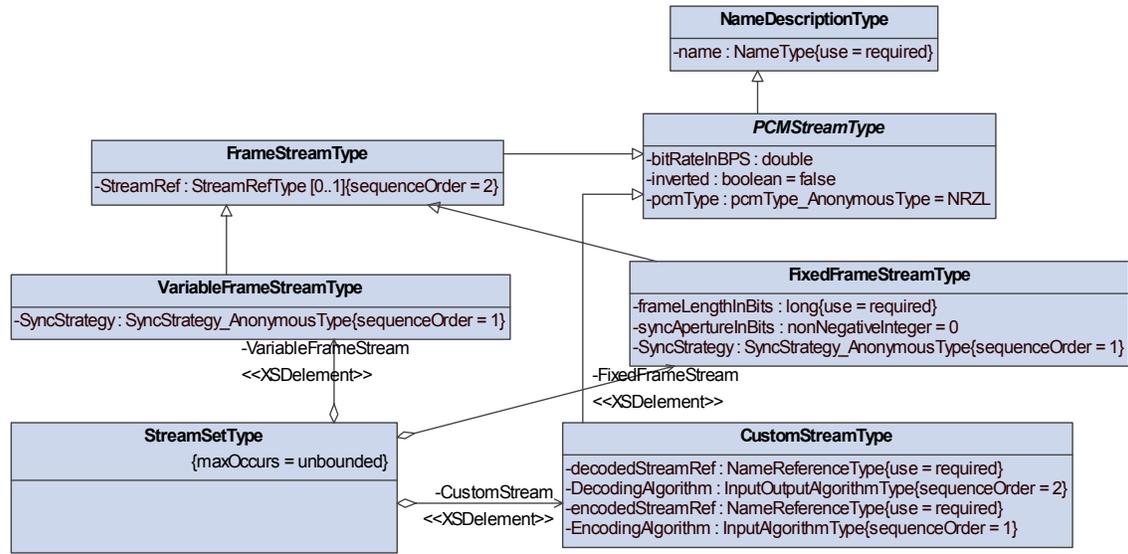


Figure 6 StreamSet

6.1.3.6 AlgorithmSet

An AlgorithmSet is an unordered collection of Algorithms. In spacecraft ground systems, it is necessary to perform some specialized processing to process the telemetry, and preprocess commands. There are a number of predefined algorithms and the algorithm section makes it possible to reference externally defined algorithms for arbitrarily sophisticated data processing.

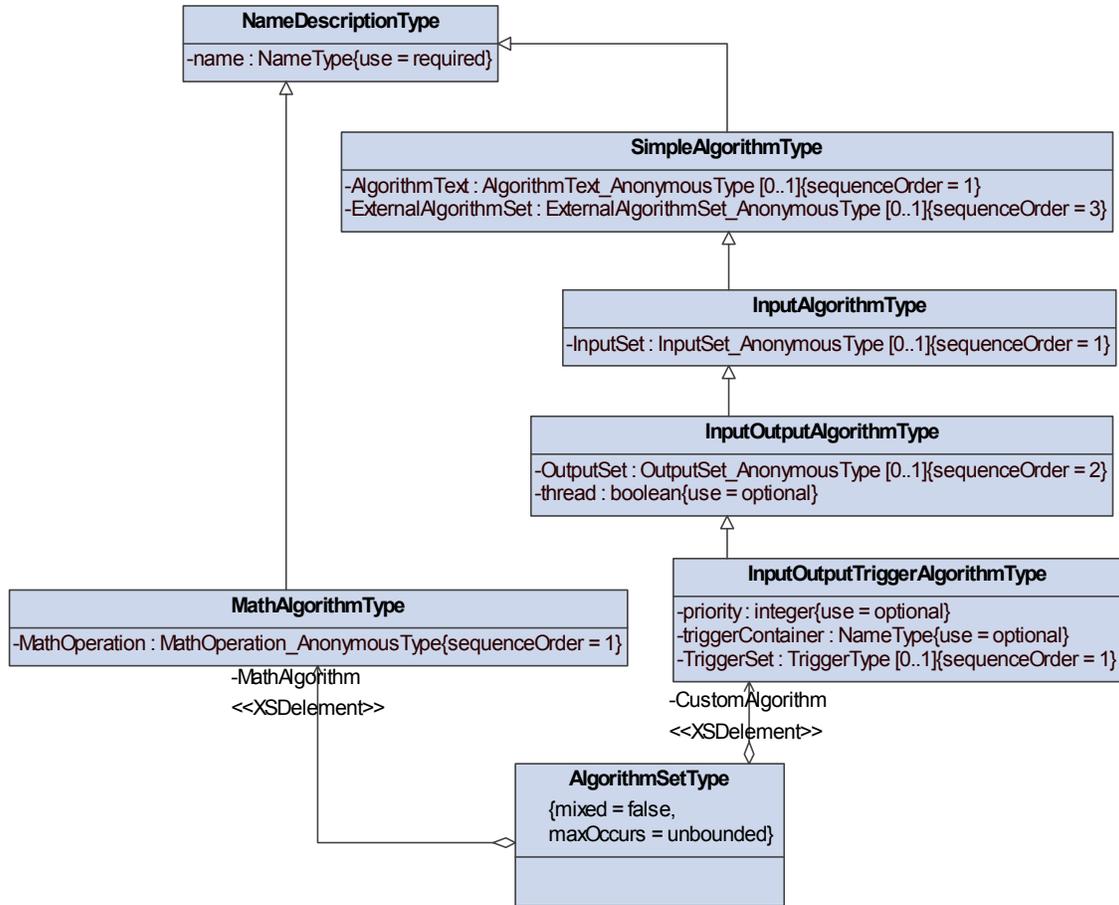


Figure 7 AlgorithmSet

6.1.3 CommandMetaData

The CommandMetaData element is very similar to TelemetryMetaData, but also contains information that is specific only to commanding. CommandMetaData has a ParameterTypeSet, a ParameterSet, a ContainerSet, a MessageSet, a StreamSet, and an AlgorithmSet – exactly like TelemetryMetaData. CommandMetaData, however, also has an ArgumentTypeSet and a MetaCommandSet.

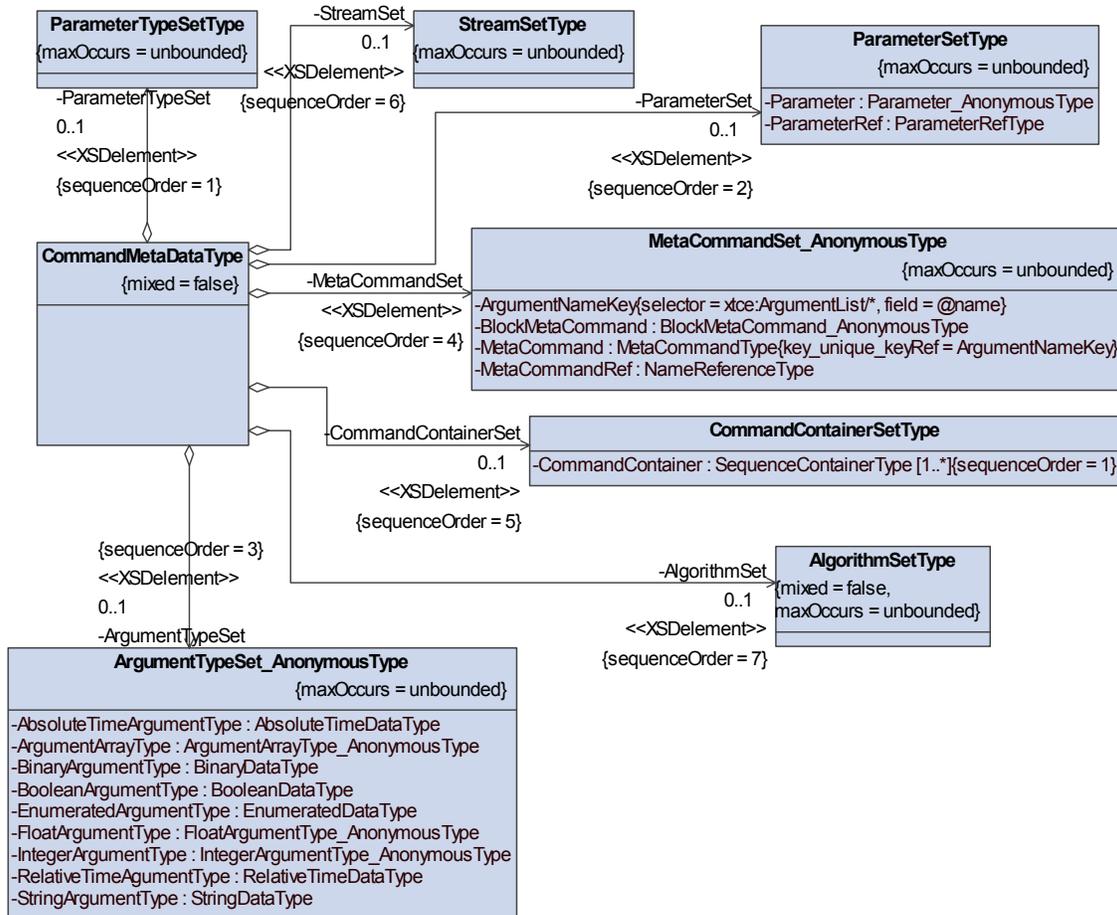


Figure 8 CommandMetaData

6.1.3.1 ArgumentTypeSet

An ArgumentTypeSet is an unordered collection of ArgumentTypes. ArgumentTypes (very similar to ParameterTypes) are the MetaData for Command Arguments; ArgumentTypes are instantiated to create Arguments. ArgumentType contains the description of something that can have a value and is used as an operator supplied option to a Command (Command Argument). Information contained in ArgumentType includes the data type, description, valid range, engineering units and string conversion specifications and calibrations. Most Arguments are sent via a data link and must also include information about how the value is encoded for transmission. This information includes size in bits, byte order, data type, and parity checks. All of the encoding information is in one of four different ‘DataEncoding’ elements. XTCE supports four different types of Encoding: IntegerDataEncoding, FloatDataEncoding, StringEncoding and BinaryDataEncoding. Note that the data encoding type only speaks to how the Command argument is transmitted, not how it is handled on the SpaceSystem or ground.

6.1.3.2 MetaCommandSet

A MetaCommandSet contains an unordered collection of MetaCommands. MetaCommands are descriptions of commands. MetaCommands have a name, a BaseMetaCommand, an ArgumentList, a CommandContainer, a TransmissionConstraintList, a DefaultSignificance, a ContextSignificanceList, an Interlock, Verifiers, and a ParameterToSetList.

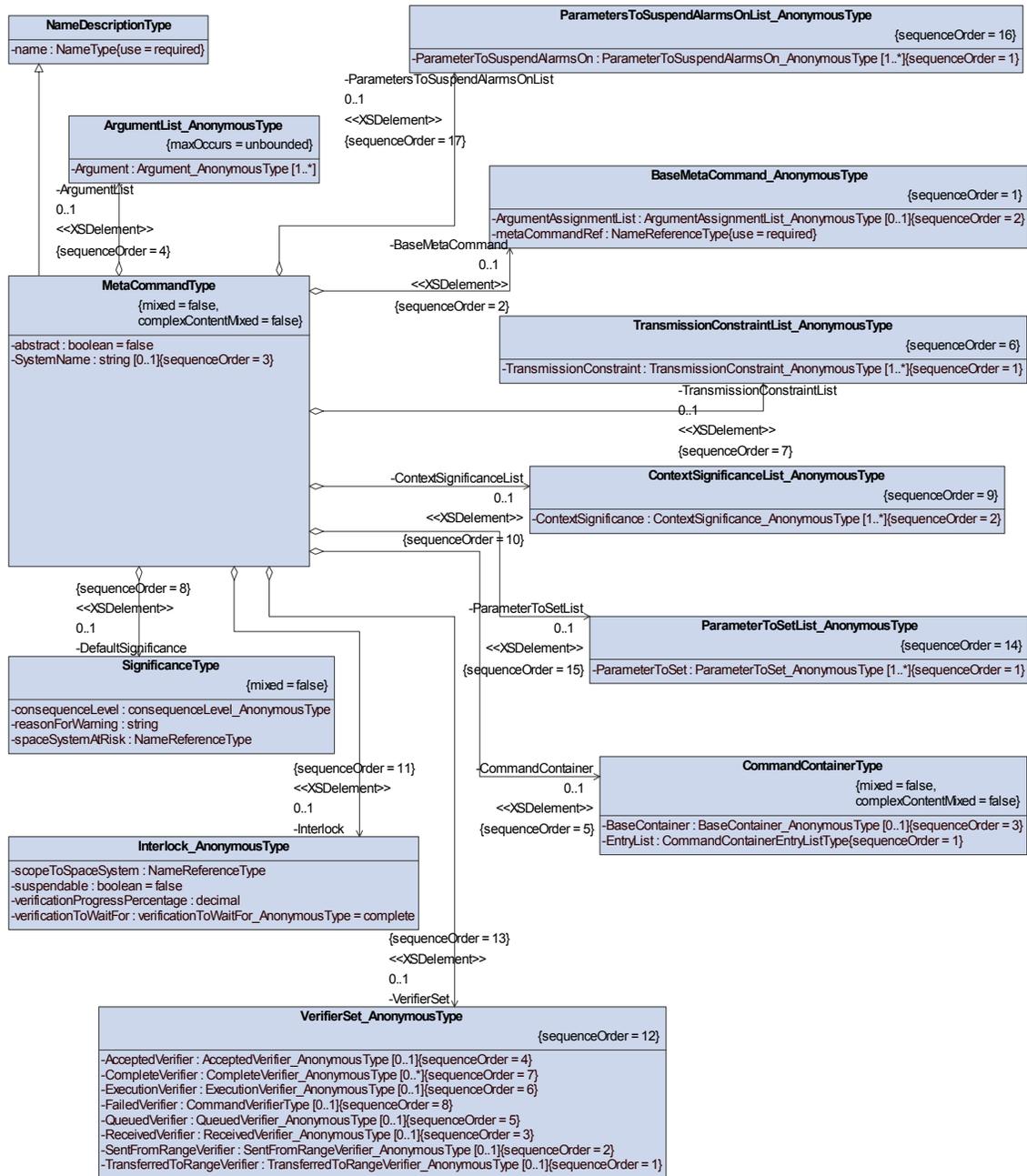


Figure 9 MetaCommandType

6.1.3.2.1 BaseMetaCommand

The MetaCommand is derived from this Base. Arguments of the base MetaCommand are further specified in this MetaCommand.

6.1.3.2.2 ArgumentList

An ArgumentList is an ordered collection of Arguments. Many commands have one or more options. These are called command arguments. Command arguments may be of any of the standard data types. MetaCommand arguments are local to the MetaCommand.

6.1.3.2.3 CommandContainer

A Command Container tells how to package this command – very similar to SequenceContainers, but CommandContainers may also have arguments and fixed values in the sequence. Each MetaCommand may have one CommandContainer.

6.1.3.2.4 TransmissionConstraintList

TransmissionConstraintList is an ordered list of TransmissionConstraints. A CommandTransmission constraint is used to check that the command can be run in the current operating mode and may block the transmission of the command if the constraint condition is true.

6.1.3.2.5 DefaultSignificance and ContextSignificanceList

Some Command and Control Systems may require special user access confirmations before transmitting commands with certain levels. The Significance includes the name of the SpaceSystem at risk, and a significance level. MetaCommands will also inherit any Significance defined in the Base MetaCommand. Significance levels are: none, watch, warning, distress, critical and severe. Additionally, it is possible to change or have different significance levels set as driven by the operating context of the SpaceSystem.

6.1.3.2.6 Interlock

An Interlock is a type of Constraint, but not on Command instances of this MetaCommand; Interlocks apply instead to the next command. An Interlock will block successive commands until this command has reached a certain stage (through verifications). Interlocks are scoped to a SpaceSystem basis.

6.1.3.2.6 Verifiers

A Command Verifier is a conditional check on the telemetry from a SpaceSystem that provides positive indication on the successful execution of a command. There are eight different verifiers each associated with difference stages in command completion: TransferredToRange, TransferredFromRange, Received, Accepted, Queued, Execution, Complete, and Failed. There may be multiple completion verifiers. Completed verifiers are added to the Base MetaCommand verifiers. All others will replace a verifier defined in a Base MetaCommand.

6.1.3.2.7 ParameterToSetList

The ParameterToSetList is an ordered collection of ParametersToSet. A ParameterToSet is a Parameter whose value will be set after the Command has reached a certain stage. New Parameters to Set are appended to the Base Command list

6.1.4 ServiceSet

ServiceSet is an unordered collection of Services. A service is a logical grouping of containers and/or messages.

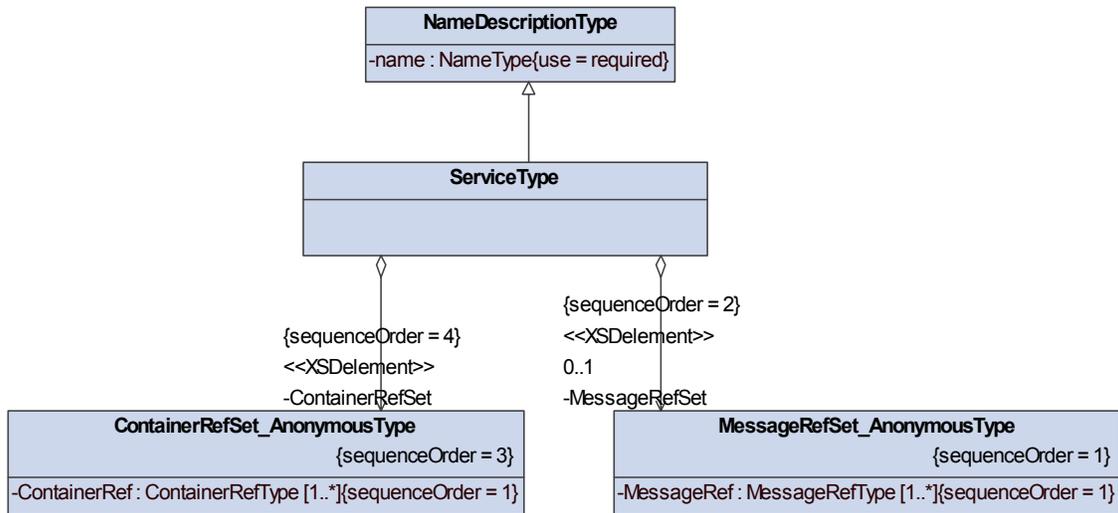


Figure 10 ServiceType

6.1.6 Defaults

Defaults has default data encoding for ParameterTypes and ArgumentTypes and default parameter time association that will be applied to all Parameters within this SpaceSystem. These defaults may be overridden by sub-SpaceSystems or by the Types or Parameters themselves. Defaults simply provides a means to avoid repeating attributes such as ‘bit order’ for every Type definition.

6.2 The Schema

The W3C XML schema is the normative specification. The schema is provided in Annex A. Any XML document compliant with this specification must validate with the schema and any other rules noted in the ‘appinfo’ annotation. Style notes used within the schema are provided in Annex B.

Annex A -The SpaceSystem Schema

A.1 Introduction

The XTCE normative specification is contained entirely as a W3C XML Schema. This schema is available as a standalone document at <http://www.omg.org/space/xtce/SpaceSystem1.1.xsd>

A.2 Schema Text

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
```

Style Notes, used throughout the schema:

- Element and Type names begin with a capital letter.
- Type names end with the word "Type".
- Attribute names begin with a lowercase letter.
- Usually, when the UML class diagram references classes, W3C Elements are used, and whenever the UML references simple types (strings, ints), W3C Attributes are used. In general, attributes are preferred over elements because they're easier to deal with in SAX and DOM, but whenever the Element/Attribute may one day carry metadata, elements should be used. One exception, is enumerated classes, because enumerations may be defined for attributes but not for elements.
- Bias toward self-describing names over short, bandwidth conserving ones.
- Use mixed case in names rather than underscores to combine multiple words (camelCase).
- A documentation annotation is included in every element and type definition. Annotations for a type are included with the type definition, use of the type is annotated in the element definition.
- Hints on units (for values with units) are provided in the names of attributes and elements (e.g. "dataRateInBPS" is preferred over "dataRate" OR "frameLengthInBits" is preferred over "frameLength").
- Major elements or any elements used multiple times are first defined with a complexType definition
- All collections are put inside either a "List" element or a "Set" Element depending on whether the collection is ordered or unordered.
- Simplicity in the XML files is favored over simplicity in the Schema
- Whenever an additional validity check must be performed that is not describable in the schema language, an appinfo annotation describes that validity check.

```
-->
<schema targetNamespace="http://www.omg.org/space/xtce" xmlns:xtce="http://www.omg.org/space/xtce"
xmlns="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" attributeFormDefault="unqualified"
version="1.1">
  <annotation>
    <documentation>OMG Document Number: dtc/2005-01-05</documentation>
    <documentation>$Id: SpaceSystemV1.1.xsd 166 2005-12-05 04:53:03Z gerry $</documentation>
    <documentation xml:lang="en">This is the master schema for the OMG Space Domain Task Force XML
Telemetric and Command data Exchange (XTCE) format.</documentation>
  </annotation>
  <!-- ***** SpaceSystem -->
  <element name="SpaceSystem" type="xtce:SpaceSystemType" nillable="true">
    <annotation>
      <documentation>The ROOT Element</documentation>
    </annotation>
    <key name="parameterNameKey">
      <annotation>
        <documentation>ensure unique parameter name at the system level</documentation>
      </annotation>
      <selector xpath="xtce:TelemetryMetaData/ParameterSet/* | xtce:CommandMetaData/ParameterSet/*"/>
      <field xpath="@name"/>
    </key>
    <key name="parameterTypeNameKey">
      <annotation>
        <documentation>ensure unique parameter type name at the system level</documentation>
      </annotation>
      <selector xpath="xtce:TelemetryMetaData/ParameterTypeSet/* |
xtce:CommandMetaData/ParameterTypeSet/*"/>
      <field xpath="@name"/>
    </key>
  </element>

```

```

</key>
<key name="metaCommandNameKey">
  <annotation>
    <documentation>ensure unique metaCommand name at the system level</documentation>
  </annotation>
  <selector xpath="xtce:MetaCommandData/MetaCommandSet/**">
    <field xpath="@name"/>
  </key>
<key name="algorithmNameKey">
  <annotation>
    <documentation>ensure unique algorithm name at the system level</documentation>
  </annotation>
  <selector xpath="xtce:TelemetryMetaData/AlgorithmSet/* | xtce:CommandMetaData/AlgorithmSet/**">
    <field xpath="@name"/>
  </key>
<key name="streamNameKey">
  <annotation>
    <documentation>ensure unique stream name at the system level</documentation>
  </annotation>
  <selector xpath="xtce:TelemetryMetaData/StreamSet/* | xtce:CommandMetaData/StreamSet/**">
    <field xpath="@name"/>
  </key>
<key name="serviceNameKey">
  <annotation>
    <documentation>ensure unique service name at the system level</documentation>
  </annotation>
  <selector xpath="xtce:ServiceSet/**">
    <field xpath="@name"/>
  </key>
<key name="containerNameKey">
  <annotation>
    <documentation>ensure container stream name at the system level</documentation>
  </annotation>
  <selector xpath="xtce:TelemetryMetaData/ContainerSet/* | xtce:CommandMetaData/ContainerSet/**">
    <field xpath="@name"/>
  </key>
<key name="messageNameKey">
  <selector xpath="xtce:TelemetryMetaData/MessageSet/**">
    <field xpath="@name"/>
  </key>
</element>
<complexType name="SpaceSystemType" mixed="false">
  <annotation>
    <documentation>SpaceSystem is a collection of SpaceSystem(s) including space assets, ground assets,
    multi-satellite systems and sub-systems. A SpaceSystem is the root element for the set of data necessary to monitor and
    command an arbitrary space device - this includes the binary decomposition the data streams going into and out of a
    device.</documentation>
  </annotation>
  <complexContent mixed="false">
    <extension base="xtce:NameDescriptionType">
      <sequence>
        <element name="Header" type="xtce:HeaderType" minOccurs="0"/>
        <element name="TelemetryMetaData" type="xtce:TelemetryMetaDataType" minOccurs="0"/>
        <element name="CommandMetaData" type="xtce:CommandMetaDataType" minOccurs="0"/>
        <element name="ServiceSet" minOccurs="0">
          <annotation>
            <documentation>A service is a logical grouping of container and/or
messages.</documentation>
          </annotation>
          <complexType>
            <sequence>
              <element name="Service" type="xtce:ServiceType" maxOccurs="unbounded"/>
            </sequence>
          </complexType>
        </element>
        <element name="Defaults" minOccurs="0">
          <annotation>
            <documentation>Defaults has default data encoding for ParameterTypes and
ArgumentTypes and default parameter time association that will be applied to all Parameters within this SpaceSystem.

```

These defaults may be overridden by sub-SpaceSystems or by the Types or Parameters themselves. Defaults simply provides a means to avoid repeating attributes such as 'bit order' for every Type definition.</documentation>

```

</annotation>
<complexType>
  <sequence>
    <element name="DefaultDataEncoding" type="xtce:DataEncodingType"
minOccurs="0"/>
    <annotation>
      <documentation>Since the data encoding (bit order and byte order) is normally
the same throughout a spacesystem, using this element allows that data encoding to be specified as a
default.</documentation>
    </annotation>
  </element>
  <element name="ParameterTimeAssociation" type="xtce:TimeAssociationType"
minOccurs="0"/>
    <annotation>
      <documentation>Default time to associate each ParameterInstance
with.</documentation>
    </annotation>
  </element>
</sequence>
</complexType>
</element>
<element ref="xtce:SpaceSystem" minOccurs="0" maxOccurs="unbounded"/>
</sequence>
</extension>
</complexContent>
</complexType>
<complexType name="CommandMetaData" mixed="false">
  <annotation>
    <documentation xml:lang="en">Command Meta Data contains information about
commands</documentation>
  </annotation>
  <sequence>
    <element name="ParameterTypeSet" type="xtce:ParameterTypeSetType" minOccurs="0"/>
      <annotation>
        <documentation>A list of parameter types</documentation>
      </annotation>
    </element>
    <element name="ParameterSet" type="xtce:ParameterSetType" minOccurs="0"/>
      <annotation>
        <documentation>Parameters referenced by MetaCommands. This Parameter Set is located here so
that MetaCommand data can be built independantly of TelemetryMetaData.</documentation>
      </annotation>
    </element>
    <element name="ArgumentTypeSet" minOccurs="0"/>
      <complexType>
        <choice maxOccurs="unbounded">
          <element name="StringArgumentType" type="xtce:StringDataType"/>
          <element name="EnumeratedArgumentType" type="xtce:EnumeratedDataType"/>
          <element name="IntegerArgumentType">
            <complexType>
              <complexContent>
                <extension base="xtce:IntegerDataType">
                  <sequence>
                    <element name="DefaultAlarm" type="xtce:NumericAlarmType"
minOccurs="0"/>
                    <element name="ContextAlarmList" minOccurs="0">
                      <complexType>
                        <sequence>
                          <element name="ContextAlarm"
type="xtce:NumericContextAlarmType" maxOccurs="unbounded"/>
                        </sequence>
                      </complexType>
                    </element>
                    <element name="ValidRangeSet" minOccurs="0">
                      <annotation>
                        <documentation>Numerical ranges that define the universe of
valid values for this argument. Used to further bound argument values inside the ValidRange for the overall Data
Type</documentation>

```

```

</annotation>
<complexType>
  <sequence>
    <element name="ValidRange" type="xtce:DecimalRangeType"
maxOccurs="unbounded"/>
  </sequence>
  <attribute name="validRangeAppliesToCalibrated" type="boolean"
default="true"/>
</complexType>
</element>
</sequence>
</extension>
</complexContent>
</complexType>
</element>
<element name="BinaryArgumentType" type="xtce:BinaryDataType"/>
<element name="FloatArgumentType">
  <complexType>
    <complexContent>
      <extension base="xtce:FloatDataType">
        <sequence>
          <element name="DefaultAlarm" type="xtce:NumericAlarmType"
minOccurs="0"/>
          <element name="ContextAlarmList" minOccurs="0">
            <complexType>
              <sequence>
                <element name="ContextAlarm"
type="xtce:NumericContextAlarmType" maxOccurs="unbounded"/>
              </sequence>
            </complexType>
          </element>
          <element name="ValidRangeSet" minOccurs="0">
            <annotation>
              <documentation>Numerical ranges that define the universe of
valid values for this argument. Used to further bound argument values inside the ValidRange for the overall Data
Type</documentation>
            </annotation>
            <complexType>
              <sequence>
                <element name="ValidRange" type="xtce:DecimalRangeType"
maxOccurs="unbounded"/>
              </sequence>
              <attribute name="validRangeAppliesToCalibrated" type="boolean"
default="true"/>
            </complexType>
          </element>
          </sequence>
        </complexContent>
      </extension>
    </complexType>
  </element>
  </choice>
</complexType>
</element>
<element name="MetaCommandSet">
  <annotation>
    <documentation>A set of Command Definitions</documentation>
  </annotation>
  <complexType>
    <choice maxOccurs="unbounded">
      <element name="MetaCommand" type="xtce:MetaCommandType">

```

```

        <annotation>
            <documentation>All commands to be sent on this mission are listed here. In addition
this area has verification and validation information</documentation>
        </annotation>
        <key name="ArgumentNameKey">
            <selector xpath="xtce:ArgumentList/**"/>
            <field xpath="@name"/>
        </key>
    </element>
    <element name="MetaCommandRef" type="xtce:NameReferenceType">
        <annotation>
            <documentation>Used to include a MetaCommand defined in another sub-system in
this sub-system.</documentation>
        </annotation>
    </element>
    <element name="BlockMetaCommand">
        <annotation>
            <documentation>BlockMetaCommands are simply a list of individual MetaCommands
that can be packaged up in a single BlockMetaCommand.</documentation>
        </annotation>
        <complexType>
            <complexContent>
                <extension base="xtce:NameDescriptionType">
                    <sequence>
                        <element name="MetaCommandStepList">
                            <complexType>
                                <sequence>
                                    <element name="MetaCommandStep"
maxOccurs="unbounded">
                                        <complexType>
                                            <sequence>
                                                <element name="ArgumentList" minOccurs="0">
                                                    <complexType>
                                                        <sequence>
                                                            <element name="Argument"
maxOccurs="unbounded">
                                                                <complexType>
                                                                    <attribute
name="name" type="string" use="required"/>
                                                                    <attribute
name="value" type="string" use="required"/>
                                                                </complexType>
                                                            </element>
                                                        </sequence>
                                                    </complexType>
                                                </element>
                                            </sequence>
                                        </complexType>
                                    </element>
                                </sequence>
                            </complexType>
                            <attribute name="metaCommandRef"
type="xtce:NameReferenceType" use="required"/>
                        </element>
                    </sequence>
                </complexType>
            </extension>
        </complexContent>
    </complexType>
</element>
</choice>
</complexType>
</element>
<element name="CommandContainerSet" type="xtce:CommandContainerSetType" minOccurs="0">
    <annotation>
        <documentation>The Command Container defines the construction of a
Command.</documentation>
    </annotation>
</element>
<element name="StreamSet" type="xtce:StreamSetType" minOccurs="0"/>
<element name="AlgorithmSet" type="xtce:AlgorithmSetType" minOccurs="0"/>

```

```

</sequence>
</complexType>
<complexType name="TelemetryMetaData" mixed="false">
  <annotation>
    <documentation>All the data about telemetry is contained in TelemetryMetaData</documentation>
  </annotation>
  <sequence>
    <element name="ParameterTypeSet" type="xtce:ParameterTypeSetType" minOccurs="0">
      <annotation>
        <documentation>A list of parameter types</documentation>
      </annotation>
    </element>
    <element name="ParameterSet" type="xtce:ParameterSetType" minOccurs="0">
      <annotation>
        <documentation>A list of Parameters for this Space System. </documentation>
      </annotation>
    </element>
    <element name="ContainerSet" type="xtce:ContainerSetType" minOccurs="0">
      <annotation>
        <documentation xml:lang="en">Holds the list of all potential container definitions for telemetry. Containers may parts of packets or TDM, and then groups of the containers, and then an entire entity -- such as a packet. In order to maximize re-used for duplication, the pieces may defined once here, and then assembled as needed into larger structures, also here.</documentation>
      </annotation>
      <key name="ContainerKey2">
        <selector xpath="Container"/>
        <field xpath="Id"/>
      </key>
    </element>
    <element name="MessageSet" minOccurs="0">
      <annotation>
        <documentation>Messages are an alternative method of uniquely identifying containers within a Service. A message provides a test in the form of MatchCriteria to match to a container. A simple example might be: [When minorframeID=21, the message is the 21st minorframe container. The collection of messages to search thru will be bound by a Service.</documentation>
      </annotation>
      <complexType>
        <sequence>
          <element name="Message" maxOccurs="unbounded">
            <complexType>
              <complexContent>
                <extension base="xtce:NameDescriptionType">
                  <sequence>
                    <element name="MatchCriteria" type="xtce:MatchCriteriaType"/>
                    <element name="ContainRef" type="xtce:ContainerRefType">
                      <annotation>
                        <documentation>The ContainerRef should point to ROOT container that will describe an entire packet/minor frame or chunk of telemetry.</documentation>
                      </annotation>
                    </element>
                  </sequence>
                </extension>
              </complexContent>
            </complexType>
          </element>
          <attribute name="name" type="string"/>
        </sequence>
      </complexType>
    </element>
    <element name="StreamSet" type="xtce:StreamSetType" minOccurs="0"/>
    <element name="AlgorithmSet" type="xtce:AlgorithmSetType" minOccurs="0"/>
  </sequence>
</complexType>
<complexType name="ServiceRefType">
  <annotation>
    <documentation xml:lang="en">A reference to a Service</documentation>
  </annotation>
  <simpleContent>
    <extension base="xtce:NameReferenceType">
      <attribute name="serviceRef" type="xtce:NameReferenceType" use="required"/>
    </extension>
  </simpleContent>

```

```

        </extension>
    </simpleContent>
</complexType>
<complexType name="AlgorithmSetType" mixed="false">
    <annotation>
        <documentation>An unordered collection of algorithms</documentation>
    </annotation>
    <choice maxOccurs="unbounded">
        <element name="CustomAlgorithm" type="xtce:InputOutputTriggerAlgorithmType"/>
        <element name="MathAlgorithm" type="xtce:MathAlgorithmType"/>
    </choice>
</complexType>
<!-- ***** End of Top Level SpaceSystem Schema -->
<!-- ***** Packaging Schema -->
<annotation>
    <documentation xml:lang="en">This schema defines the dictionary for containers, which in turn describe the
physical composition of data in a communication system</documentation>
</annotation>
<complexType name="ContainerType" abstract="true" mixed="false">
    <annotation>
        <documentation xml:lang="en">An abstract block of data; used as the base type for more specific container
types</documentation>
    </annotation>
    <complexContent mixed="false">
        <extension base="xtce:NameDescriptionType">
            <sequence>
                <annotation>
                    <documentation>RateInStream is used to: a) generate alarms when the Container is updated
too frequently or too infrequently, b) provide some 'guidelines' for generating forward link containers, c) provide some
guidelines for spacecraft simulators to generate telemetry containers. If necessary, these rates may be defined on a per
stream basis.</documentation>
                    <appinfo>The software should check that any Stream names referenced in the RateInStreamSet
actually exist.</appinfo>
                </annotation>
                <element name="DefaultRateInStream" type="xtce:RateInStreamType" minOccurs="0"/>
                <element name="RateInStreamSet" minOccurs="0">
                    <complexType>
                        <sequence>
                            <element name="RateInStream" maxOccurs="unbounded">
                                <complexType>
                                    <complexContent>
                                        <extension base="xtce:RateInStreamType">
                                            <attribute name="streamRef" type="xtce:NameReferenceType"
use="required"/>
                                        </extension>
                                    </complexContent>
                                </complexType>
                            </element>
                        </sequence>
                    </complexType>
                </element>
            </sequence>
        </extension>
    </complexContent>
</complexType>
<complexType name="BinaryEncoding" type="xtce:BinaryDataEncodingType" minOccurs="0">
    <annotation>
        <documentation>May be used to indicate error detection and correction, chage byte order,
provide the size (when it can't be derived), or perform some custom processing.</documentation>
    </annotation>
</element>
</sequence>
</extension>
</complexContent>
</complexType>
<complexType name="SequenceContainerType">
    <annotation>
        <documentation>A list of raw parameters, parameter segments, stream segments, containers, or container
segments. Sequence containers may inherit from other sequence containers; when they do, the sequence in the parent
SequenceContainer is 'inherited' and if the location of entries in the child sequence is not specified, it is assumed to start
where the parent sequence ended. Parent sequence containers may be marked as "abstract". The idle pattern is part of
any unallocated space in the Container.</documentation>
    </annotation>

```

```

<complexContent>
  <extension base="xtce:ContainerType">
    <sequence>
      <element name="EntryList" type="xtce:EntryListType"/>
      <element name="BaseContainer" minOccurs="0">
        <complexType>
          <sequence>
            <element name="RestrictionCriteria">
              <annotation>
                <documentation>Given that this Container is the Base container type,
RestrictionCriteria lists conditions that must be true for this Container to be 'this' subContainer type. May be a simple
Comparison List, a Boolean Expression, and/or in a Graph of containers established by the
NextContainer</documentation>
              </annotation>
            </complexType>
            <complexContent>
              <extension base="xtce:MatchCriteriaType">
                <choice>
                  <element name="NextContainer" type="xtce:ContainerRefType"
minOccurs="0"/>
                </choice>
              </extension>
            </complexContent>
          </complexType>
        </element>
      </sequence>
      <attribute name="containerRef" type="xtce:NameReferenceType" use="required"/>
    </complexType>
  </element>
</sequence>
<attribute name="abstract" type="boolean"/>
<attribute name="idlePattern" type="xtce:FixedIntegerValueType" default="0x0"/>
</extension>
</complexContent>
</complexType>
<complexType name="SequenceEntryType">
  <annotation>
    <documentation>An abstract type used by sequence containers. An entry contains a location in the
container. The location may be either fixed or dynamic, absolute (to the start or end of the enclosing container, or relative
to either the previous or subsequent entry). Entries may also repeat.</documentation>
  </annotation>
  <sequence>
    <element name="LocationInContainerInBits" minOccurs="0">
      <annotation>
        <documentation>If no LocationInContainer value is given, the entry is assumed to begin immediately
after the previous entry.</documentation>
      </annotation>
      <complexType>
        <complexContent>
          <extension base="xtce:IntegerValueType">
            <attribute name="referenceLocation" default="previousEntry">
              <annotation>
                <documentation>The location may be relative to the start of the container
(containerStart), relative to the end of the previous entry (previousEntry), relative to the end of the container
(containerEnd), or relative to the entry that follows this one (nextEntry). If going forward (containerStart and
previousEntry) then, then the location refers to the start of the Entry. If going backwards (containerEnd and nextEntry)
then, the location refers to the end of the entry.</documentation>
              </annotation>
            </simpleType>
            <restriction base="string">
              <enumeration value="containerStart"/>
              <enumeration value="containerEnd"/>
              <enumeration value="previousEntry"/>
              <enumeration value="nextEntry"/>
            </restriction>
          </simpleType>
        </attribute>
      </extension>
    </complexContent>
  </complexType>
</complexType>

```

```

</element>
<element name="RepeatEntry" type="xtce:RepeatType" minOccurs="0">
  <annotation>
    <documentation xml:lang="en">May be used when this entry repeats itself in the sequence
container. If not supplied, the entry does not repeat.</documentation>
  </annotation>
</element>
<element name="IncludeCondition" type="xtce:MatchCriteriaType" minOccurs="0">
  <annotation>
    <documentation>This entry will only be included in the sequence when this condition is true. If no
IncludeCondition is given, then it will be included. A parameter that is not included will be treated as if it did not exist in
the sequence at all.</documentation>
  </annotation>
</element>
</sequence>
</complexType>
<complexType name="ContainerRefType">
  <annotation>
    <documentation xml:lang="en">Holds a reference to a container</documentation>
  </annotation>
  <attribute name="containerRef" type="xtce:NameReferenceType" use="required">
    <annotation>
      <documentation xml:lang="en">name of container</documentation>
    </annotation>
  </attribute>
</complexType>
<complexType name="MessageRefType">
  <annotation>
    <documentation xml:lang="en">Holds a reference to a message</documentation>
  </annotation>
  <attribute name="messageRef" type="xtce:NameReferenceType" use="required">
    <annotation>
      <documentation xml:lang="en">name of message</documentation>
    </annotation>
  </attribute>
</complexType>
<complexType name="ServiceType">
  <annotation>
    <documentation xml:lang="en">Holds a set of services, logical groups of containers OR messages (not
both).</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:NameDescriptionType">
      <sequence>
        <element name="MessageRefSet" minOccurs="0">
          <complexType>
            <sequence>
              <element name="MessageRef" type="xtce:MessageRefType"
maxOccurs="unbounded"/>
            </sequence>
          </complexType>
        </element>
        <element name="ContainerRefSet">
          <complexType>
            <sequence>
              <element name="ContainerRef" type="xtce:ContainerRefType"
maxOccurs="unbounded"/>
            </sequence>
          </complexType>
        </element>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<complexType name="ContainerSetType">
  <annotation>
    <documentation>Unordered Set of Containers</documentation>
  </annotation>
  <choice maxOccurs="unbounded">
    <element name="SequenceContainer" type="xtce:SequenceContainerType">

```

```

        <annotation>
          <documentation>SequenceContainers define sequences of parameters or other containers.
        </documentation>
      </annotation>
    </element>
  </choice>
</complexType>
<complexType name="EntryListType" mixed="false">
  <annotation>
    <documentation>Contains an ordered list of Entries. Used in Sequence Container</documentation>
  </annotation>
  <choice minOccurs="0" maxOccurs="unbounded">
    <element name="ParameterRefEntry" type="xtce:ParameterRefEntryType"/>
    <element name="ParameterSegmentRefEntry" type="xtce:ParameterSegmentRefEntryType"/>
    <element name="ContainerRefEntry" type="xtce:ContainerRefEntryType"/>
    <element name="ContainerSegmentRefEntry" type="xtce:ContainerSegmentRefEntryType"/>
    <element name="StreamSegmentEntry" type="xtce:StreamSegmentEntryType"/>
    <element name="IndirectParameterRefEntry" type="xtce:IndirectParameterRefEntryType"/>
    <element name="ArrayParameterRefEntry" type="xtce:ArrayParameterRefEntryType"/>
  </choice>
</complexType>
<complexType name="ParameterRefEntryType">
  <annotation>
    <documentation>An entry that is a single Parameter</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:SequenceEntryType">
      <attribute name="parameterRef" type="xtce:NameReferenceType" use="required"/>
    </extension>
  </complexContent>
</complexType>
<complexType name="ParameterSegmentRefEntryType">
  <annotation>
    <documentation>An entry that is only a portion of a parameter value indicating that the entire parameter
    value must be assembled from other parameter segments. It is assumed that parameter segments happen sequentially
    in time, that is the first part if a telemetry parameter first, however (and there's always a however), if this is not the case
    the order of this parameter segment may be supplied with the order attribute where the first segment
    order="0".</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:SequenceEntryType">
      <attribute name="parameterRef" type="xtce:NameReferenceType" use="required"/>
      <attribute name="order" type="positiveInteger"/>
      <attribute name="sizeInBits" type="positiveInteger" use="required"/>
    </extension>
  </complexContent>
</complexType>
<complexType name="ContainerRefEntryType">
  <annotation>
    <documentation>An entry that is simply a reference to another container.</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:SequenceEntryType">
      <attribute name="containerRef" type="xtce:NameReferenceType" use="required"/>
    </extension>
  </complexContent>
</complexType>
<complexType name="ContainerSegmentRefEntryType">
  <annotation>
    <documentation>An entry that is only a portion of a container indicating that the entire container must be
    assembled from other container segments. It is assumed that container segments happen sequentially in time, that is the
    first part of a container is first, however (and there's always a however), if this is not the case the order of this container
    segment may be supplied with the order attribute where the first segment order="0". Each instance of a container cannot
    overlap in the overall sequence with another instance</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:SequenceEntryType">
      <attribute name="containerRef" type="xtce:NameReferenceType" use="required"/>
      <attribute name="order" type="positiveInteger"/>
      <attribute name="sizeInBits" type="positiveInteger" use="required"/>
    </extension>
  </complexContent>

```

```

        </extension>
    </complexContent>
</complexType>
<complexType name="StreamSegmentEntryType">
    <annotation>
        <documentation>An entry that is a portion of a stream (streams are by definition, assumed continuous) It is assumed that stream segments happen sequentially in time, that is the first part if a steam first, however, if this is not the case the order of the stream segments may be supplied with the order attribute where the first segment order="0".</documentation>
    </annotation>
    <complexContent>
        <extension base="xtce:SequenceEntryType">
            <attribute name="streamRef" type="xtce:NameReferenceType" use="required"/>
            <attribute name="order" type="positiveInteger"/>
            <attribute name="sizeInBits" type="positiveInteger" use="required"/>
        </extension>
    </complexContent>
</complexType>
<complexType name="IndirectParameterRefEntryType">
    <annotation>
        <documentation>An entry whose name is given by the value of a ParamameterInstance. This entry may be used to implement dwell telemetry streams. The value of the parameter in ParameterInstance must use either the name of the Parameter or its alias. If it's an alias name, the alias namespace is supplied as an attribute.</documentation>
    </annotation>
    <complexContent>
        <extension base="xtce:SequenceEntryType">
            <sequence>
                <element name="ParameterInstance" type="xtce:ParameterInstanceRefType"/>
            </sequence>
            <attribute name="aliasNameSpace" type="string"/>
        </extension>
    </complexContent>
</complexType>
<complexType name="ArrayParameterRefEntryType">
    <annotation>
        <documentation>An entry that is an array parameter. This entry is somewhat special because the entry may represent only a part of the Array and it's important to decrbe which diminsions of the array come first in the sequence as well as the size of the array.</documentation>
    </annotation>
    <complexContent>
        <extension base="xtce:SequenceEntryType">
            <sequence>
                <element name="DimensionList">
                    <annotation>
                        <documentation>Where the Dimension list is in this form: Array[1stDim][2ndDim][lastDim]. The last dimension is assumed to be the least significant - that is this dimension will cycle through it's combination before the next to last dimension changes. The order MUST ascend or the array will need to be broken out entry by entry.</documentation>
                    </annotation>
                    <complexType>
                        <sequence>
                            <element name="Dimension" maxOccurs="unbounded">
                                <annotation>
                                    <documentation>For partial entries of an array, the starting and ending index for each dimension, OR the Size must be specified. Indexes are zero based.</documentation>
                                </annotation>
                                <appinfo>For an ArrayParameterType of size N, their should be N Dimensions</appinfo>
                                <appinfo>An array made up by multiple Entries should not have index's that overlap, but should be continuous.</appinfo>
                            </element>
                        </sequence>
                    </complexType>
                </element>
            </sequence>
        </extension>
    </complexContent>

```

```

        </element>
      </sequence>
    </complexType>
  </element>
</sequence>
<attribute name="parameterRef" type="xtce:NameReferenceType" use="required"/>
<attribute name="lastEntryForThisArrayInstance" type="boolean" default="false"/>
</extension>
</complexContent>
</complexType>
<complexType name="RateInStreamType">
  <annotation>
    <documentation>Used in packaging to define the expected rate that any individual container will be in a
Stream</documentation>
  </annotation>
  <attribute name="basis" default="perSecond">
    <simpleType>
      <restriction base="string">
        <enumeration value="perSecond"/>
        <enumeration value="perContainerUpdate"/>
      </restriction>
    </simpleType>
  </attribute>
  <attribute name="minimumValue" type="double"/>
  <attribute name="maximumValue" type="double"/>
</complexType>
<!-- ***** End of Packaging Schema -->
<!-- ***** Telemetry Schema -->
<complexType name="ParameterPropertiesType" mixed="false">
  <annotation>
    <documentation>A wrapper for those properties that are unique to telemetry parameters.</documentation>
  </annotation>
  <sequence>
    <element name="SystemName" type="string" minOccurs="0">
      <annotation>
        <documentation>Optional. Normally used when the database is built in a flat, non-hierarchical
format</documentation>
      </annotation>
    </element>
    <element name="ValidityCondition" type="xtce:MatchCriteriaType" minOccurs="0">
      <annotation>
        <documentation>Optional condition that must be true for this Parameter to be valid</documentation>
      </annotation>
    </element>
    <element name="PhysicalAddressSet" minOccurs="0">
      <annotation>
        <documentation>One or more physical addresses may be associated with each Parameter.
Examples of physical addresses include a location on the spacecraft or a location on a data collection bus.
</documentation>
      </annotation>
    </complexType>
    <sequence>
      <element name="PhysicalAddress" minOccurs="0" maxOccurs="unbounded">
        <annotation>
          <documentation xml:lang="en">Contains the address (e.g., channel information)
required to process the spacecraft telemetry streams. May be an onboard id, a mux address, or a physical
location.</documentation>
          <documentation xml:lang="en">Contains the address (channel information) required to
process the spacecraft telemetry streams</documentation>
        </annotation>
      </complexType>
    </sequence>
  </element>
</complexType>
</element>

```

```

<element name="TimeAssociation" type="xtce:TimeAssociationType" minOccurs="0">
  <annotation>
    <documentation>This time will override any Default value for TimeAssociation. </documentation>
  </annotation>
</element>
</sequence>
<attribute name="dataSource" use="optional">
  <annotation>
    <documentation>A telemetered Parameter is one that will have values in telemetry. A derived
Parameter is one that is calculated, usually be an Algorithm. A constant Parameter is one that is used as a constant in
the system (e.g. a vehicle id). A local Parameter is one that is used purely on the ground (e.g. a ground command
counter).</documentation>
  </annotation>
  <simpleType>
    <restriction base="string">
      <enumeration value="telemetered"/>
      <enumeration value="derived"/>
      <enumeration value="constant"/>
      <enumeration value="local"/>
    </restriction>
  </simpleType>
</attribute>
<attribute name="readOnly" type="boolean" use="optional" default="false">
  <annotation>
    <documentation>A Parameter marked as 'readOnly' true is constant and non-settable</documentation>
  </annotation>
</attribute>
</complexType>
<complexType name="TimeAssociationType">
  <annotation>
    <documentation>Telemetry parameter instances are oftentimes "time-tagged" with a timing signal either
provided on the ground or on the space system. This data element allows one to specify which of possibly many
AbsoluteTimeParameters to use to "time-tag" parameter instances with. </documentation>
    <appinfo>The parameter ref must be to an AbsoluteTime Parameter</appinfo>
  </annotation>
  <simpleContent>
    <extension base="xtce:ParameterInstanceRefType">
      <attribute name="interpolateTime" type="boolean" default="true">
        <annotation>
          <documentation xml:lang="en">If true, then the current value of the AbsoluteTime will be
projected to current time. I.E., if the value of the AbsoluteTime parameter was set 10 seconds ago, then 10 seconds will
be added to it's value before associating this time with the parameter.</documentation>
        </annotation>
      </attribute>
      <attribute name="offset" type="date">
        <annotation>
          <documentation>The offset is used to supply a relative time offset from the time association and
to this parameter</documentation>
        </annotation>
      </attribute>
    </extension>
  </simpleContent>
</complexType>
<complexType name="ParameterInstanceRefType">
  <annotation>
    <documentation>A reference to an instance of a Parameter. Used when the value of a parameter is
required for a calculation or as an index value. A positive value for instance is forward in time, a negative value for count
is backward in time, a 0 value for count means use the current value of the parameter or the first value in a
container.</documentation>
  </annotation>
  <simpleContent>
    <extension base="xtce:ParameterRefType">
      <attribute name="instance" type="integer" default="0"/>
      <attribute name="useCalibratedValue" type="boolean" default="true"/>
    </extension>
  </simpleContent>
</complexType>
<complexType name="ParameterRefType">
  <annotation>

```

<documentation xml:lang="en">A reference to a Parameter. Uses Unix 'like' naming across the SpaceSystem Tree (e.g., SimpleSat/Bus/EPDS/BatteryOne/Voltage). To reference an individual member of an array use the zero based bracket notation commonly used in languages like C, C++, and Java.</documentation>

```

</annotation>
<simpleContent>
  <extension base="xtce:NameReferenceType">
    <attribute name="parameterRef" type="xtce:NameReferenceType" use="required"/>
  </extension>
</simpleContent>
</complexType>
<complexType name="PhysicalAddressType" mixed="false">
  <annotation>
    <documentation>When it's important to know the physical address(s) on the spacecraft that this parameter
may be collected from, use this. </documentation>
  </annotation>
  <sequence>
    <element name="SubAddress" type="xtce:PhysicalAddressType" minOccurs="0"/>
  </sequence>
  <attribute name="sourceName" type="string"/>
  <attribute name="sourceAddress" type="string"/>
</complexType>
<complexType name="ParameterTypeSetType">
  <annotation>
    <documentation xml:lang="en">Holds the list of parameter definitions. A Parameter is a description of
something that can have a value; it is not the value itself. </documentation>
  </annotation>
  <choice maxOccurs="unbounded">
    <element name="StringParameterType">
      <complexType>
        <complexContent>
          <extension base="xtce:StringDataType">
            <sequence>
              <element name="DefaultAlarm" type="xtce:StringAlarmType" minOccurs="0"/>
              <element name="ContextAlarmList" minOccurs="0">
                <complexType>
                  <sequence>
                    <element name="ContextAlarm" type="xtce:StringAlarmType"/>
                  </sequence>
                </complexType>
              </element>
            </sequence>
          </extension>
        </complexContent>
      </complexType>
    </element>
    <element name="EnumeratedParameterType">
      <complexType>
        <complexContent>
          <extension base="xtce:EnumeratedDataType">
            <sequence>
              <element name="DefaultAlarm" type="xtce:EnumerationAlarmType" minOccurs="0"/>
              <element name="ContextAlarmList" minOccurs="0">
                <complexType>
                  <sequence>
                    <element name="ContextAlarm">
                      <complexType>
                        <complexContent>
                          <extension base="xtce:EnumerationAlarmType">
                            <sequence>
                              <element name="ContextMatch"
type="xtce:MatchCriteriaType"/>
                            </sequence>
                          </extension>
                        </complexContent>
                      </complexType>
                    </element>
                  </sequence>
                </complexType>
              </element>
            </sequence>
          </extension>
        </complexContent>
      </complexType>
    </element>
  </choice>
</complexType>
</sequence>

```

```

        </extension>
    </complexContent>
</complexType>
</element>
<element name="IntegerParameterType">
    <complexType>
        <complexContent>
            <extension base="xtce:IntegerDataType">
                <sequence>
                    <element name="DefaultAlarm" type="xtce:NumericAlarmType" minOccurs="0"/>
                    <element name="ContextAlarmList" minOccurs="0">
                        <complexType>
                            <sequence>
                                <element name="ContextAlarm" type="xtce:NumericContextAlarmType"
maxOccurs="unbounded"/>
                            </sequence>
                        </complexType>
                    </element>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</element>
<element name="BinaryParameterType">
    <complexType>
        <complexContent>
            <extension base="xtce:BinaryDataType">
                <sequence>
                    <element name="DefaultAlarm" type="xtce:AlarmType" minOccurs="0"/>
                    <element name="ContextAlarmList" minOccurs="0">
                        <complexType>
                            <sequence>
                                <element name="ContextAlarm">
                                    <complexType>
                                        <complexContent>
                                            <extension base="xtce:AlarmType">
                                                <sequence>
                                                    <element name="ContextMatch"
type="xtce:MatchCriteriaType"/>
                                                </sequence>
                                            </extension>
                                        </complexContent>
                                    </complexType>
                                </element>
                            </sequence>
                        </complexType>
                    </element>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</element>
<element name="FloatParameterType">
    <complexType>
        <complexContent>
            <extension base="xtce:FloatDataType">
                <sequence>
                    <element name="DefaultAlarm" type="xtce:NumericAlarmType" minOccurs="0"/>
                    <element name="ContextAlarmList" minOccurs="0">
                        <complexType>
                            <sequence>
                                <element name="ContextAlarm" type="xtce:NumericContextAlarmType"
maxOccurs="unbounded"/>
                            </sequence>
                        </complexType>
                    </element>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</element>

```

```

</element>
<element name="BooleanParameterType">
  <complexType>
    <complexContent>
      <extension base="xtce:BooleanDataType">
        <sequence>
          <element name="DefaultAlarm" type="xtce:BooleanAlarmType" minOccurs="0"/>
          <element name="ContextAlarmList" minOccurs="0">
            <complexType>
              <sequence>
                <element name="ContextAlarm">
                  <complexType>
                    <complexContent>
                      <extension base="xtce:BooleanAlarmType">
                        <sequence>
                          <element name="ContextMatch"
type="xtce:MatchCriteriaType"/>
                        </sequence>
                      </extension>
                    </complexContent>
                  </complexType>
                </element>
              </sequence>
            </complexType>
          </element>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
</element>
<element name="RelativeTimeParameterType">
  <complexType>
    <complexContent>
      <extension base="xtce:RelativeTimeDataType">
        <sequence>
          <element name="DefaultAlarm" type="xtce:TimeAlarmType" minOccurs="0"/>
          <element name="ContextAlarmList" minOccurs="0">
            <complexType>
              <sequence>
                <element name="ContextAlarm" type="xtce:TimeContextAlarmType"
maxOccurs="unbounded"/>
              </sequence>
            </complexType>
          </element>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
</element>
<element name="AbsoluteTimeParameterType" type="xtce:AbsoluteTimeDataType"/>
<element name="ArrayParameterType">
  <annotation>
    <documentation>An array type. Will be an array of parameters of the type referenced in
'arrayTypeRef' and have the number of array dimensions as specified in 'numberOfDimensions' </documentation>
  </annotation>
  <complexType>
    <complexContent>
      <extension base="xtce:NameDescriptionType">
        <attribute name="arrayTypeRef" type="xtce:NameReferenceType" use="required"/>
        <attribute name="numberOfDimensions" type="positiveInteger" use="required"/>
      </extension>
    </complexContent>
  </complexType>
</element>
</choice>
</complexType>
<!-- ***** End of Telemetry Schema -->
<!-- ***** -->
<!-- ***** Command Schema -->
<!--CommandDefinitionType -->

```

```

<complexType name="MetaCommandType" mixed="false">
  <annotation>
    <documentation xml:lang="en">A type definition used as the base type for a
CommandDefinition</documentation>
  </annotation>
  <complexContent mixed="false">
    <extension base="xtce:NameDescriptionType">
      <sequence>
        <element name="BaseMetaCommand" minOccurs="0">
          <annotation>
            <documentation>The MetaCommand is derived from this Base. Arguments of the base
MetaCommand are further specified.</documentation>
          </annotation>
          <complexType>
            <sequence>
              <element name="ArgumentAssignmentList" minOccurs="0">
                <complexType>
                  <sequence>
                    <element name="ArgumentAssignment" maxOccurs="unbounded">
                      <complexType>
                        <attribute name="argumentName"
type="xtce:NameReferenceType" use="required"/>
                        <attribute name="argumentValue" type="string" use="required"/>
                      </complexType>
                    </element>
                  </sequence>
                </complexType>
              </element>
            </sequence>
            <attribute name="metaCommandRef" type="xtce:NameReferenceType" use="required"/>
          </complexType>
        </element>
        <element name="SystemName" type="string" minOccurs="0">
          <annotation>
            <documentation>Optional. Normally used when the database is built in a flat, non-
hierarchical format</documentation>
          </annotation>
        </element>
        <element name="ArgumentList" minOccurs="0">
          <annotation>
            <documentation>Many commands have one or more options. These are called command
arguments. Command arguments may be of any of the standard data types. MetaCommand arguments are local to the
MetaCommand.</documentation>
          </annotation>
          <complexType>
            <choice maxOccurs="unbounded">
              <element name="Argument" maxOccurs="unbounded">
                <annotation>
                  <appinfo>Need to ensure that the named types actually exist</appinfo>
                </annotation>
                <complexType>
                  <complexContent>
                    <extension base="xtce:NameDescriptionType">
                      <attribute name="argumentTypeRef" type="xtce:NameReferenceType"
use="required"/>
                    </extension>
                  </complexContent>
                </complexType>
              </element>
            </choice>
          </complexType>
        </element>
        <element name="CommandContainer" type="xtce:CommandContainerType" minOccurs="0">
          <annotation>
            <documentation>Tells how to package this command</documentation>
          </annotation>
        </element>
        <element name="TransmissionConstraintList" minOccurs="0">
          <annotation>

```

```

        <documentation>Appended to the TransmissionConstraint List of the base command.
Constraints are checked in order. </documentation>
    </annotation>
    <complexType>
        <sequence>
            <element name="TransmissionConstraint" maxOccurs="unbounded">
                <annotation>
                    <documentation>A CommandTransmission constraint is used to check that the
command can be run in the current operating mode and may block the transmission of the command if the constraint
condition is true.</documentation>
                </annotation>
                <complexType>
                    <complexContent>
                        <extension base="xtce:MatchCriteriaType">
                            <attribute name="timeOut" type="xtce:RelativeTimeType">
                                <annotation>
                                    <documentation>Pause during timeOut, fail when the timeout
passes</documentation>
                                </annotation>
                                <!-- removed for CASTOR: default="PT0S" -->
                            </attribute>
                            <attribute name="suspendable" type="boolean" default="false">
                                <annotation>
                                    <documentation>Indicates whether the constraints for a
Command may be suspended.</documentation>
                                </annotation>
                            </attribute>
                        </extension>
                    </complexContent>
                </complexType>
            </element>
        </sequence>
    </complexType>
</element>
<element name="DefaultSignificance" type="xtce:SignificanceType" minOccurs="0">
    <annotation>
        <documentation>Some Command and Control Systems may require special user access or
confirmations before transmitting commands with certain levels. The level is inherited from the Base
MetaCommand.</documentation>
    </annotation>
</element>
<element name="ContextSignificancelist" minOccurs="0">
    <annotation>
        <documentation>Used when the significance (possible consequence) of a command varies
by the operating context</documentation>
    </annotation>
    <complexType>
        <sequence>
            <element name="ContextSignificance" maxOccurs="unbounded">
                <complexType>
                    <sequence>
                        <element name="ContextMatch" type="xtce:MatchCriteriaType"/>
                        <element name="Significance" type="xtce:SignificanceType"/>
                    </sequence>
                </complexType>
            </element>
        </sequence>
    </complexType>
</element>
<element name="Interlock" minOccurs="0">
    <annotation>
        <documentation>An Interlock is a type of Constraint, but not on Command instances of this
MetaCommand; Interlocks apply instead to the next command. An Interlock will block successive commands until this
command has reached a certain stage (through verifications). Interlocks are scoped to a SpaceSystem
basis.</documentation>
    </annotation>
    <complexType>
        <attribute name="scopeToSpaceSystem" type="xtce:NameReferenceType">
            <annotation>

```

<documentation>The name of a SpaceSystem this Interlock applies to. By default, it only applies to the SpaceSystem that contains this MetaCommand.</documentation>

```

</annotation>
</attribute>
<attribute name="verificationToWaitFor" default="complete">
  <simpleType>
    <restriction base="string">
      <enumeration value="transferredToRange"/>
      <enumeration value="sentFromRange"/>
      <enumeration value="received"/>
      <enumeration value="accepted"/>
      <enumeration value="queued"/>
      <enumeration value="executing"/>
      <enumeration value="complete"/>
    </restriction>
  </simpleType>
</attribute>
<attribute name="verificationProgressPercentage" type="decimal">
  <annotation>

```

<documentation>Only applies when the verificationToWaitFor attribute is 'queued' or 'executing'.</documentation>

```

</annotation>
</attribute>
<attribute name="suspendable" type="boolean" default="false">
  <annotation>

```

<documentation>A flag that indicates that under special circumstances, this Interlock can be suspended.</documentation>

```

</annotation>
</attribute>
</complexType>
</element>
<element name="VerifierSet" minOccurs="0">
  <annotation>

```

<documentation>A Command Verifier is a conditional check on the telemetry from a SpaceSystem that provides positive indication on the successful execution of a command. Completed verifiers are added to the Base MetaCommand verifiers. All others will replace a verifier defined in a Base MetaCommand.</documentation>

```

</annotation>
<complexType>
  <sequence>
    <element name="TransferredToRangeVerifier" minOccurs="0">
      <annotation>

```

<documentation>Transferred to range means the command has been received by a the network that connects the ground system to the spacecraft. Obviously, this verifier must come from something other than the spacecraft.</documentation>

```

</annotation>
<complexType>
  <complexContent>
    <extension base="xtce:CommandVerifierType"/>
  </complexContent>
</complexType>
</element>
<element name="SentFromRangeVerifier" minOccurs="0">
  <annotation>

```

<documentation>Sent from range means the command has been transmitted to the spacecraft by a the network that connects the ground system to the spacecraft. Obviously, this verifier must come from something other than the spacecraft.</documentation>

```

</annotation>
<complexType>
  <complexContent>
    <extension base="xtce:CommandVerifierType"/>
  </complexContent>
</complexType>
</element>
<element name="ReceivedVerifier" minOccurs="0">
  <annotation>

```

<documentation>A verifier that simply means the SpaceSystem has received the command.</documentation>

```

</annotation>
<complexType>
  <complexContent>

```

```

        <extension base="xtce:CommandVerifierType"/>
    </complexContent>
</complexType>
</element>
<element name="AcceptedVerifier" minOccurs="0">
    <annotation>
        <documentation>A verifier that means the SpaceSystem has accepted the
command</documentation>
    </annotation>
</complexType>
    <complexContent>
        <extension base="xtce:CommandVerifierType"/>
    </complexContent>
</complexType>
</element>
<element name="QueuedVerifier" minOccurs="0">
    <annotation>
        <documentation>A verifier that means the command is scheduled for
execution by the SpaceSystem.</documentation>
    </annotation>
</complexType>
    <complexContent>
        <extension base="xtce:CommandVerifierType"/>
    </complexContent>
</complexType>
</element>
<element name="ExecutionVerifier" minOccurs="0">
    <annotation>
        <documentation>A verifier that indicates that the command is being executed.
An optional Element indicates how far along the command has progressed either as a fixed value or an (possibly scaled)
ParameterInstance value.</documentation>
    </annotation>
</complexType>
    <complexContent>
        <extension base="xtce:CommandVerifierType">
            <sequence minOccurs="0">
                <element name="PercentComplete"
type="xtce:DecimalValueType"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>
</element>
<element name="CompleteVerifier" minOccurs="0" maxOccurs="unbounded">
    <annotation>
        <documentation>A possible set of verifiers that all must be true for the
command be considered completed.</documentation>
    </annotation>
</complexType>
    <complexContent>
        <extension base="xtce:CommandVerifierType">
            <sequence minOccurs="0">
                <element name="ReturnParmRef"
type="xtce:ParameterRefType"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>
</element>
<element name="FailedVerifier" type="xtce:CommandVerifierType" minOccurs="0">
    <annotation>
        <documentation>When true, indicates that the command failed. timeToWait is
how long to wait for the FailedVerifier to test true.</documentation>
    </annotation>
</element>
</sequence>
</complexType>
</element>
<element name="ParameterToSetList" minOccurs="0">
    <annotation>

```

```

        <documentation>Parameters that are set with a new value after the command has been
sent. Appended to the Base Command list</documentation>
    </annotation>
    <complexType>
    <sequence>
    <element name="ParameterToSet" maxOccurs="unbounded">
    <annotation>
    <documentation>Sets a Parameter to a new value after the command has been
verified (all verifications have passed)</documentation>
    </annotation>
    <complexType>
    <complexContent>
    <extension base="xtce:ParameterToSetType">
    <attribute name="verifierToTriggerOn" default="release">
    <simpleType>
    <restriction base="string">
    <enumeration value="release"/>
    <enumeration value="transferredToRange"/>
    <enumeration value="sentFromRange"/>
    <enumeration value="received"/>
    <enumeration value="accepted"/>
    <enumeration value="queued"/>
    <enumeration value="executing"/>
    <enumeration value="complete"/>
    <enumeration value="failed"/>
    </restriction>
    </simpleType>
    </attribute>
    </extension>
    </complexContent>
    </complexType>
    </element>
    </sequence>
    </complexType>
    </element>
    <element name="ParametersToSuspendAlarmsOnList" minOccurs="0">
    <annotation>
    <documentation>Sometimes it is necessary to suspend alarms (particularly 'change' alarms
for commands that will change the value of a Parameter</documentation>
    </annotation>
    <complexType>
    <sequence>
    <element name="ParameterToSuspendAlarmsOn" maxOccurs="unbounded">
    <annotation>
    <documentation>Will suspend all Alarms associated with this Parameter for the
given suspense time after the given verifier</documentation>
    </annotation>
    <complexType>
    <simpleContent>
    <extension base="xtce:NameReferenceType">
    <attribute name="suspenseTime" type="xtce:RelativeTimeType"
use="required"/>
    <attribute name="verifierToTriggerOn" default="release">
    <simpleType>
    <restriction base="string">
    <enumeration value="release"/>
    <enumeration value="transferredToRange"/>
    <enumeration value="sentFromRange"/>
    <enumeration value="received"/>
    <enumeration value="accepted"/>
    <enumeration value="queued"/>
    <enumeration value="executing"/>
    <enumeration value="complete"/>
    <enumeration value="failed"/>
    </restriction>
    </simpleType>
    </attribute>
    </extension>
    </simpleContent>
    </complexType>
    </element>
    </sequence>
    </complexType>

```

```

        </element>
      </sequence>
    </complexType>
  </element>
</sequence>
<attribute name="abstract" type="boolean" default="false"/>
</extension>
</complexContent>
</complexType>
<complexType name="CommandContainerEntryListType" mixed="false">
  <annotation>
    <documentation>Similar to an EntryList type but also may include command arguments or -as a convenience
- fixed value entries.</documentation>
  </annotation>
  <choice minOccurs="0" maxOccurs="unbounded">
    <element name="ParameterRefEntry" type="xtce:ParameterRefEntryType"/>
    <element name="ParameterSegmentRefEntry" type="xtce:ParameterSegmentRefEntryType"/>
    <element name="ContainerRefEntry" type="xtce:ContainerRefEntryType"/>
    <element name="ContainerSegmentRefEntry" type="xtce:ContainerSegmentRefEntryType"/>
    <element name="StreamSegmentEntry" type="xtce:StreamSegmentEntryType"/>
    <element name="IndirectParameterRefEntry" type="xtce:IndirectParameterRefEntryType"/>
    <element name="ArrayParameterRefEntry" type="xtce:ArrayParameterRefEntryType"/>
    <element name="ArgumentRefEntry">
      <complexType>
        <complexContent>
          <extension base="xtce:SequenceEntryType">
            <attribute name="argumentRef" type="xtce:NameReferenceType" use="required"/>
          </extension>
        </complexContent>
      </complexType>
    </element>
    <element name="ArrayArgumentRefEntry" type="xtce:ArrayParameterRefEntryType"/>
    <element name="FixedValueEntry">
      <complexType>
        <complexContent>
          <extension base="xtce:SequenceEntryType">
            <attribute name="binaryValue" type="hexBinary" use="required"/>
            <attribute name="sizeInBits" type="integer"/>
          </extension>
        </complexContent>
      </complexType>
    </element>
  </choice>
</complexType>
<complexType name="CommandContainerType" mixed="false">
  <annotation>
    <documentation>The Key = Command Op Code. Each MetaCommand may have one CommandContainer.
The sequence may now contain command fields</documentation>
  </annotation>
  <complexContent mixed="false">
    <extension base="xtce:ContainerType">
      <sequence>
        <element name="EntryList" type="xtce:CommandContainerEntryListType"/>
        <element name="BaseContainer" minOccurs="0">
          <complexType>
            <sequence>
              <element name="RestrictionCriteria" minOccurs="0">
                <annotation>
                  <documentation>Given that this Container is the Base container type,
RestrictionCriteria lists conditions that must be true for this Container to be 'this' subContainer type. May be a simple
Comparison List, a Boolean Expression, and/or in a Graph of containers established by the
NextContainer</documentation>
                </annotation>
              </complexType>
            <complexContent>
              <extension base="xtce:MatchCriteriaType">
                <choice>
                  <element name="NextContainer" type="xtce:ContainerRefType"
minOccurs="0"/>
                </choice>
              </extension>
            </complexContent>
          </complexType>
        </element>
      </sequence>
    </extension>
  </complexContent>

```

```

        </extension>
      </complexContent>
    </complexType>
  </element>
</sequence>
<attribute name="containerRef" type="xtce:NameReferenceType" use="required"/>
</complexType>
</element>
</sequence>
</extension>
</complexContent>
</complexType>
<complexType name="CommandVerifierType">
  <annotation>
    <documentation>A command verifier is used to check that the command has been successfully executed.
Command Verifiers may be either a Custom Algorithm or a Boolean Check or the presence of a Container for a relative
change in the value of a Parameter. The CheckWindow is a time period where the verification must test true to
pass.</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:OptionalNameDescriptionType">
      <sequence>
        <choice>
          <element name="ComparisonList">
            <annotation>
              <documentation>All comparisons must be true</documentation>
            </annotation>
            <complexType>
              <sequence>
                <element name="Comparison" type="xtce:ComparisonType"
maxOccurs="unbounded"/>
              </sequence>
            </complexType>
          </element>
          <element name="ContainerRef" type="xtce:ContainerRefType">
            <annotation>
              <documentation>When verification is the existence of a Container</documentation>
            </annotation>
          </element>
          <element name="ParameterValueChange">
            <annotation>
              <documentation>Used to look for relative change in a Parameter value. Only useful for
numeric Parameters</documentation>
            </annotation>
            <complexType>
              <sequence>
                <element name="ParameterRef" type="xtce:ParameterRefType"/>
                <element name="Change">
                  <complexType>
                    <attribute name="value" type="decimal" use="required"/>
                  </complexType>
                </element>
              </sequence>
            </complexType>
          </element>
          <element name="CustomAlgorithm" type="xtce:InputAlgorithmType"/>
          <element name="BooleanExpression" type="xtce:BooleanExpressionType"/>
          <element name="Comparison" type="xtce:ComparisonType"/>
        </choice>
        <choice>
          <element name="CheckWindow">
            <complexType>
              <attribute name="timeToStartChecking" type="xtce:RelativeTimeType"/>
              <attribute name="timeToStopChecking" type="xtce:RelativeTimeType" use="required"/>
              <attribute name="timeWindowsRelativeTo" default="timeLastVerifierPassed">
                <simpleType>
                  <restriction base="string">
                    <enumeration value="commandRelease"/>
                    <enumeration value="timeLastVerifierPassed"/>
                  </restriction>
                </simpleType>
              </attribute>
            </complexType>
          </element>
        </choice>
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

```

        </simpleType>
      </attribute>
    </complexType>
  </element>
  <element name="CheckWindowAlgorithms">
    <annotation>
      <documentation>Used when times must be calculated</documentation>
    </annotation>
    <complexType>
      <sequence>
        <element name="StartCheck" type="xtce:InputAlgorithmType"/>
        <element name="StopTime" type="xtce:InputAlgorithmType"/>
      </sequence>
    </complexType>
  </element>
</choice>
</sequence>
</extension>
</complexContent>
</complexType>
<complexType name="CommandVerifierType2">
  <annotation>
    <documentation>A command verifier is used to check that the command has been successfully executed.
    Command Verifiers may be either a Custom Algorithm or a Boolean Check or the presence of a Container for a relative
    change in the value of a Parameter. The CheckWindow is a time period where the verification must test true to
    pass.</documentation>
  </annotation>
  <sequence>
    <choice>
      <element name="ComparisonList">
        <annotation>
          <documentation>All comparisons must be true</documentation>
        </annotation>
        <complexType>
          <sequence>
            <element name="Comparison" type="xtce:ComparisonType" maxOccurs="unbounded"/>
          </sequence>
        </complexType>
      </element>
      <element name="ContainerRef" type="xtce:ContainerRefType">
        <annotation>
          <documentation>When verification is the existence of a Container</documentation>
        </annotation>
      </element>
      <element name="ParameterValueChange">
        <annotation>
          <documentation>Used to look for relative change in a Parameter value. Only usefull for numeric
          Parameters</documentation>
        </annotation>
        <complexType>
          <sequence>
            <element name="ParameterRef" type="xtce:ParameterRefType"/>
            <element name="Change">
              <complexType>
                <attribute name="value" type="decimal" use="required"/>
              </complexType>
            </element>
          </sequence>
        </complexType>
      </element>
      <element name="CustomAlgorithm" type="xtce:InputAlgorithmType"/>
      <element name="BooleanExpression" type="xtce:BooleanExpressionType"/>
      <element name="Comparison" type="xtce:ComparisonType"/>
    </choice>
  </choice>
  <choice>
    <element name="CheckWindow">
      <complexType>
        <attribute name="timeToStartChecking" type="xtce:RelativeTimeType"/>
        <attribute name="timeToStopChecking" type="xtce:RelativeTimeType" use="required"/>
        <attribute name="timeWindowIsRelativeTo" default="timeLastVerifierPassed">

```

```

        <simpleType>
          <restriction base="string">
            <enumeration value="commandRelease"/>
            <enumeration value="timeLastVerifierPassed"/>
          </restriction>
        </simpleType>
      </attribute>
    </complexType>
  </element>
  <element name="CheckWindowAlgorithms">
    <annotation>
      <documentation>Used when times must be calculated</documentation>
    </annotation>
    <complexType>
      <sequence>
        <element name="StartCheck" type="xtce:InputAlgorithmType"/>
        <element name="StopTime" type="xtce:InputAlgorithmType"/>
      </sequence>
    </complexType>
  </element>
</choice>
</sequence>
</complexType>
<complexType name="ParameterToSetType">
  <annotation>
    <documentation>Used by Meta Command to indicate ground Parameters that should be set after completion
of a command. </documentation>
  </annotation>
  <sequence>
    <element name="ParameterRef" type="xtce:ParameterRefType"/>
    <element name="Derivation" type="xtce:MathOperationType"/>
  </sequence>
</complexType>
<complexType name="CommandContainerSetType">
  <annotation>
    <documentation>Contains an unordered Set of Command Containers</documentation>
  </annotation>
  <sequence>
    <element name="CommandContainer" type="xtce:SequenceContainerType" maxOccurs="unbounded"/>
  </sequence>
</complexType>
<complexType name="SignificanceType" mixed="false">
  <annotation>
    <documentation>Significance provides some cautionary information about the potential consequence of each
MetaCommand.</documentation>
  </annotation>
  <attribute name="spaceSystemAtRisk" type="xtce:NameReferenceType">
    <annotation>
      <documentation>If none is supplied, then the current SpaceSystem is assumed to be the one at risk by
the issuance of this command</documentation>
    </annotation>
  </attribute>
  <attribute name="reasonForWarning" type="string"/>
  <attribute name="consequenceLevel">
    <annotation>
      <documentation>No specific meanings have been assigned to these different levels, but they mirror the
Alarm levels of Telemetry.</documentation>
    </annotation>
  </attribute>
  <simpleType>
    <restriction base="string">
      <enumeration value="none"/>
      <enumeration value="watch"/>
      <enumeration value="warning"/>
      <enumeration value="distress"/>
      <enumeration value="critical"/>
      <enumeration value="severe"/>
    </restriction>
  </simpleType>
</attribute>
</complexType>

```

```

<complexType name="RelativeTimeDataType">
  <annotation>
    <documentation>Used to contain a relative time value. Used to describe a relative time. Normally used for
time offsets. A Relative time is expressed as PnYn MnDTnH nMnS, where nY represents the number of years, nM the
number of months, nD the number of days, 'T' is the date/time separator, nH the number of hours, nM the number of
minutes and nS the number of seconds. The number of seconds can include decimal digits to arbitrary precision. For
example, to indicate a duration of 1 year, 2 months, 3 days, 10 hours, and 30 minutes, one would write:
P1Y2M3DT10H30M. One could also indicate a duration of minus 120 days as: -P120D. An extension of Schema duration
type. </documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:BaseTimeDataType"/>
  </complexContent>
</complexType>
<!-- ***** End of Command Definition Schema -->
<!-- *****</comment>
<!-- ***** Algorithm Schema -->
<annotation>
  <documentation xml:lang="en">This schema defines the structure for an Algorithm. An Algorithm may be one of
a growing set of pre-defined algorithms or a named escape into a user defined algorithm where (depending on the
system) the name of the algorithm may be a java class, a function in a shared library, an external program or some other
reference to an outside algorithm. At some later date, this schema may also allow the logic of the user defined algorithm
to be defined within the instance document itself (perhaps using MathML?).</documentation>
</annotation>
<complexType name="SimpleAlgorithmType">
  <annotation>
    <documentation xml:lang="en">The simplest form of algorithm, a SimpleAlgorithmType contains an area for
a free-form pseudo code description of the algorithm plus a Set of references to external algorithms. External algorithms
are usually unique to a ground system type. Multiple external algorithms are possible because XTCE documents may be
used across multiple ground systems.</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:NameDescriptionType">
      <sequence>
        <element name="AlgorithmText" minOccurs="0">
          <annotation>
            <documentation xml:lang="en">This optional element may be used to enter Pseudo or
actual code for the algorithm. The language for the algorithm is specified with the language attribute</documentation>
          </annotation>
          <complexType>
            <simpleContent>
              <extension base="string">
                <attribute name="language" type="string" default="pseudo"/>
              </extension>
            </simpleContent>
          </complexType>
        </element>
        <element name="ExternalAlgorithmSet" minOccurs="0">
          <complexType>
            <sequence>
              <element name="ExternalAlgorithm" maxOccurs="unbounded">
                <annotation>
                  <documentation>This is the external algorithm. Multiple entries are provided so
that the same database may be used for multiple implementation s</documentation>
                </annotation>
                <complexType>
                  <attribute name="implementationName" type="string" use="required"/>
                  <attribute name="algorithmLocation" type="string" use="required"/>
                </complexType>
              </element>
            </sequence>
          </complexType>
        </element>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<complexType name="InputAlgorithmType">
  <annotation>
    <documentation>A set of labeled inputs is added to the SimpleAlgorithmType</documentation>
  </annotation>

```

```

</annotation>
<complexContent>
  <extension base="xtce:SimpleAlgorithmType">
    <sequence>
      <element name="InputSet" minOccurs="0">
        <complexType>
          <choice maxOccurs="unbounded">
            <element name="ParameterInstanceRef">
              <annotation>
                <documentation>Names an input parameter to the algorithm. There are two
attributes to InputParm, inputName and parameterName. parameterName is a parameter reference name for a parameter
that will be used in this algorithm. inputName is an optional "friendly" name for the input parameter. </documentation>
              </annotation>
              <complexType>
                <simpleContent>
                  <extension base="xtce:ParameterInstanceRefType">
                    <attribute name="inputName" type="string"/>
                  </extension>
                </simpleContent>
              </complexType>
            </element>
            <element name="Constant" minOccurs="0">
              <annotation>
                <documentation xml:lang="en">Names and provides a value for a constant
input to the algorithm. There are two attributes to Constant, constantName and value. constantName is a variable name
in the algorithm to be executed. value is the value of the constant to be used.</documentation>
              </annotation>
              <complexType>
                <attribute name="constantName" type="string"/>
                <attribute name="value" type="string" use="required"/>
              </complexType>
            </element>
          </choice>
        </complexType>
      </element>
    </sequence>
  </extension>
</complexContent>
</complexType>
<complexType name="InputOutputAlgorithmType">
  <annotation>
    <documentation>A set of labeled outputs are added to the SimpleInputAlgorithmType</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:InputAlgorithmType">
      <sequence>
        <element name="OutputSet" minOccurs="0">
          <complexType>
            <sequence>
              <element name="OutputParameterRef" maxOccurs="unbounded">
                <annotation>
                  <documentation>Names an output parameter to the algorithm. There are two
attributes to OutputParm, outputName and parameterName. parameterName is a parameter reference name for a
parameter that will be updated by this algorithm. outputName is an optional "friendly" name for the output
parameter.</documentation>
                </annotation>
                <complexType>
                  <simpleContent>
                    <extension base="xtce:ParameterRefType">
                      <attribute name="outputName" type="string"/>
                    </extension>
                  </simpleContent>
                </complexType>
              </element>
            </sequence>
          </complexType>
        </element>
      </sequence>
      <attribute name="thread" type="boolean" use="optional"/>
    </extension>
  </complexContent>

```

```

    </complexContent>
  </complexType>
  <complexType name="InputOutputTriggerAlgorithmType">
    <annotation>
      <documentation>A set of labeled triggers is added to the SimpleInputOutputAlgorithmType</documentation>
    </annotation>
    <complexContent>
      <extension base="xtce:InputOutputAlgorithmType">
        <sequence>
          <element name="TriggerSet" type="xtce:TriggerType" minOccurs="0"/>
        </sequence>
        <attribute name="triggerContainer" type="xtce:NameType" use="optional">
          <annotation>
            <documentation xml:lang="en">First telemetry container from which the output parameter
should be calculated.</documentation>
          </annotation>
        </attribute>
        <attribute name="priority" type="integer" use="optional">
          <annotation>
            <documentation xml:lang="en">Algorithm processing priority.</documentation>
          </annotation>
        </attribute>
      </extension>
    </complexContent>
  </complexType>
  <complexType name="CalibratorType">
    <annotation>
      <documentation xml:lang="en">Calibrators are normally used to convert to and from bit compacted numerical
data</documentation>
    </annotation>
    <complexContent>
      <extension base="xtce:OptionalNameDescriptionType">
        <choice>
          <element name="SplineCalibrator">
            <annotation>
              <documentation xml:lang="en">A calibration type where a segmented line in a raw vs
calibrated plane is described using a set of points. Raw values are converted to calibrated values by finding a position on
the line coorresponding to the raw value. The algorithm triggers on the input parameter.</documentation>
            </annotation>
            <complexType>
              <sequence>
                <element name="SplinePoint" type="xtce:SplinePointType" minOccurs="2"
maxOccurs="unbounded"/>
              </sequence>
              <attribute name="order" type="positiveInteger" default="1"/>
              <attribute name="extrapolate" type="boolean" default="false"/>
            </complexType>
          </element>
          <element name="PolynomialCalibrator" type="xtce:PolynomialType">
            <annotation>
              <documentation>A calibration type where a curve in a raw vs calibrated plane is described
using a set of polynomial coefficients. Raw values are converted to calibrated values by finding a position on the curve
corresponding to the raw value. The first coefficient belongs with the X^0 term, the next coefficient belongs to the X^1
term and so on. </documentation>
            </annotation>
          </element>
        </choice>
      </extension>
    </complexContent>
  </complexType>
  <complexType name="MathAlgorithmType">
    <annotation>
      <documentation>A simple mathematical operation</documentation>
    </annotation>
    <complexContent>
      <extension base="xtce:NameDescriptionType">
        <sequence>
          <element name="MathOperation">
            <complexType>
              <complexContent>

```

```

        <extension base="xtce:MathOperationType">
            <sequence>
                <element name="TriggerSet" type="xtce:TriggerType"/>
            </sequence>
            <attribute name="outputParameterRef" type="xtce:NameReferenceType"
use="required"/>
        </extension>
    </complexContent>
</complexType>
</element>
</sequence>
</extension>
</complexContent>
</complexType>
<complexType name="TriggerType">
    <annotation>
        <documentation>A trigger is used to initiate the processing of some algorithm. A trigger may be based on an
update of a Parameter or on a time basis. Triggers may also have a rate that limits their firing to a 1/rate
basis.</documentation>
    </annotation>
    <choice maxOccurs="unbounded">
        <element name="ParameterRef" minOccurs="0">
            <annotation>
                <documentation>Names a parameter that upon change will start the execution of the algorithm.
Holds a parameter reference name for a parameter that when it changes, will cause this algorithm to be
executed.</documentation>
            </annotation>
            <complexType>
                <simpleContent>
                    <extension base="xtce:ParameterRefType">
                        <attribute name="triggerName" type="string"/>
                    </extension>
                </simpleContent>
            </complexType>
        </element>
        <element name="TriggerFrequency" type="xtce:RelativeTimeType" minOccurs="0"/>
    </choice>
    <attribute name="name" type="string" use="optional"/>
    <attribute name="triggerRate" type="nonNegativeInteger" use="optional" default="1"/>
</complexType>
<!-- ***** End of Algorithm Schema -->
<!-- ***** Stream Definitions Schema -->
<annotation>
    <documentation xml:lang="en">This schema provides a language for defining binary stream
data.</documentation>
</annotation>
<complexType name="FrameStreamType">
    <annotation>
        <documentation xml:lang="en">The top level type definition for all data streams that are frame
based.</documentation>
    </annotation>
    <complexContent>
        <extension base="xtce:PCMStreamType">
            <sequence>
                <choice>
                    <element name="ContainerRef" type="xtce:ContainerRefType">
                        <annotation>
                            <documentation>This Container (usually abstract) is the container that is in the fixed
frame stream. Normally, this is an generalcontainer type from which many specific containers are
inherited.</documentation>
                        </annotation>
                    </element>
                    <element name="ServiceRef" type="xtce:ServiceRefType"/>
                </choice>
                <element name="StreamRef" type="xtce:StreamRefType" minOccurs="0">
                    <annotation>
                        <documentation>This is a reference to a connecting stream - say a custom
stream.</documentation>
                    </annotation>
                </element>
            </sequence>
        </extension>
    </complexContent>
</complexType>

```

```

        </sequence>
      </extension>
    </complexContent>
  </complexType>
  <complexType name="FixedFrameStreamType">
    <annotation>
      <documentation xml:lang="en">For streams that contain a series of frames with a fixed frame length where
the frames are found by looking for a marker in the data. This marker is sometimes called the frame sync pattern and
sometimes the Asynchronous Sync Marker (ASM). This marker need not be contiguous although it usually
is.</documentation>
    </annotation>
    <complexContent>
      <extension base="xtce:FrameStreamType">
        <sequence>
          <element name="SyncStrategy">
            <complexType>
              <complexContent>
                <extension base="xtce:SyncStrategyType">
                  <sequence>
                    <element name="SyncPattern">
                      <annotation>
                        <documentation xml:lang="en">The pattern of bits used to look for
frame synchronization.</documentation>
                      </annotation>
                      <complexType>
                        <attribute name="pattern" type="hexBinary" use="required">
                          <annotation>
                            <documentation>CCSDS ASM for non-turbocoded frames =
1acffc1d</documentation>
                          </annotation>
                        </attribute>
                        <attribute name="bitLocationFromStartOfContainer" type="integer"
default="0"/>
                        <attribute name="mask" type="hexBinary"/>
                        <attribute name="maskLengthInBits" type="positiveInteger">
                          <annotation>
                            <documentation>truncate the mask from the
left</documentation>
                          </annotation>
                        </attribute>
                        <attribute name="patternLengthInBits" type="positiveInteger"
use="required">
                          <annotation>
                            <documentation>truncate the pattern from the
left</documentation>
                          </annotation>
                        </attribute>
                      </complexType>
                    </element>
                  </sequence>
                </extension>
              </complexContent>
            </complexType>
          </element>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
  <complexType name="VariableFrameStreamType">
    <annotation>
      <documentation xml:lang="en">For streams that contain a series of frames with a variable frame length
where the frames are found by looking for a series of one's or zero's (usually one's). The series is called the flag. in the
PCM stream that are usually made to be illegal in the PCM stream by zero or one bit insertion. </documentation>
    </annotation>

```

```

<complexContent>
  <extension base="xtce:FrameStreamType">
    <sequence>
      <element name="SyncStrategy">
        <complexType>
          <complexContent>
            <extension base="xtce:SyncStrategyType">
              <sequence>
                <element name="Flag">
                  <annotation>
                    <documentation xml:lang="en">The pattern of bits used to look for
frame synchronization.</documentation>
                  </annotation>
                <complexType>
                  <attribute name="flagSizeInBits" type="positiveInteger" default="6"/>
                  <attribute name="flagBitType" default="ones">
                    <simpleType>
                      <restriction base="string">
                        <enumeration value="zeros"/>
                        <enumeration value="ones"/>
                      </restriction>
                    </simpleType>
                  </attribute>
                </complexType>
              </element>
            </sequence>
          </extension>
        </complexContent>
      </element>
    </sequence>
  </extension>
</complexContent>
<complexType name="CustomStreamType">
  <annotation>
    <documentation xml:lang="en">A stream type where some level of custom processing (e.g. convolutional,
encryption, compression) is performed. Has a reference to external algorithms for encoding and decoding
algorithms.</documentation>
    <appinfo>Must check to ensure that the attributes encodedStreamRef and decodedStreamRef point to valid
Streams</appinfo>
  </annotation>
  <complexContent>
    <extension base="xtce:PCMStreamType">
      <sequence>
        <element name="EncodingAlgorithm" type="xtce:InputAlgorithmType"/>
        <element name="DecodingAlgorithm" type="xtce:InputOutputAlgorithmType">
          <annotation>
            <documentation>Algorithm outputs may be used to set decoding quality
parameters.</documentation>
          </annotation>
        </element>
      </sequence>
      <attribute name="encodedStreamRef" type="xtce:NameReferenceType" use="required"/>
      <attribute name="decodedStreamRef" type="xtce:NameReferenceType" use="required"/>
    </extension>
  </complexContent>
</complexType>
<complexType name="PCMStreamType" abstract="true">
  <annotation>
    <documentation xml:lang="en">A PCM Stream Type is the high level definition for all Pulse Code Modulated
(PCM) (i.e., binary) streams.</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:NameDescriptionType">
      <attribute name="bitRateInBPS" type="double"/>
      <attribute name="pcmType" default="NRZL">
        <simpleType>
          <restriction base="string">
            <enumeration value="NRZL"/>
          </restriction>
        </simpleType>
      </attribute>
    </extension>
  </complexContent>

```

```

        <enumeration value="NRZM"/>
        <enumeration value="NRZS"/>
        <enumeration value="BiPhaseL"/>
        <enumeration value="BiPhaseM"/>
        <enumeration value="BiPhaseS"/>
    </restriction>
</simpleType>
</attribute>
<attribute name="inverted" type="boolean" default="false"/>
</extension>
</complexContent>
</complexType>
<complexType name="StreamRefType">
    <annotation>
        <documentation xml:lang="en">Holds a reference to a stream</documentation>
    </annotation>
    <attribute name="streamRef" type="xtce:NameReferenceType" use="required">
        <annotation>
            <documentation xml:lang="en">name of reference stream</documentation>
        </annotation>
    </attribute>
</complexType>
<complexType name="StreamSetType">
    <annotation>
        <documentation>Contains an unordered set of Streams.</documentation>
    </annotation>
    <choice maxOccurs="unbounded">
        <element name="FixedFrameStream" type="xtce:FixedFrameStreamType"/>
        <element name="VariableFrameStream" type="xtce:VariableFrameStreamType"/>
        <element name="CustomStream" type="xtce:CustomStreamType"/>
    </choice>
</complexType>
<complexType name="SyncStrategyType">
    <annotation>
        <documentation>A Sync Strategy specifies the strategy on how to find frames within a stream of PCM data.
        The sync strategy is based upon a state machine that begins in the 'Search' state until the first sync marker is found.
        Then it goes into the 'Verify' state until a specified number of successive good sync markers are found. Then, the state
        machine goes into the 'Lock' state, in the 'Lock' state frames are considered good. Should a sync marker be missed in
        the 'Lock' state, the state machine will transition into the 'Check' state, if the next sync marker is where it's expected within
        a specified number of frames, then the state machine will transition back to the 'Lock' state, if not it will transition back to
        'Search'. </documentation>
    </annotation>
    <sequence>
        <element name="AutoInvert" minOccurs="0">
            <annotation>
                <documentation xml:lang="en">After serching for the frame sync marker for some number of bits, it
                may be desirable to invert the incoming data, and then look for frame sync. In some cases this will require an external
                algorithm</documentation>
            </annotation>
            <complexType>
                <sequence>
                    <element name="InvertAlgorithm" type="xtce:InputAlgorithmType" minOccurs="0"/>
                </sequence>
                <attribute name="badFramesToAutoInvert" type="positiveInteger" default="1024"/>
            </complexType>
        </element>
    </sequence>
    <attribute name="verifyToLockGoodFrames" type="nonNegativeInteger" default="4"/>
    <attribute name="checkToLockGoodFrames" type="nonNegativeInteger" default="1"/>
    <attribute name="maxBitErrorsInSyncPattern" type="nonNegativeInteger" default="0">
        <annotation>
            <documentation>Maximum number of bit errors in the sync pattern (marker).</documentation>
        </annotation>
    </attribute>
</complexType>
<!-- ***** End of Stream Definition Schema -->
<!-- ***** -->
<!-- ***** DataTypes-->
<complexType name="AbsoluteTimeDataType">
    <annotation>

```

<documentation>Used to contain an absolute time. Contains an absolute (to a known epoch) time. Use the [ISO 8601] extended format CCYY-MM-DDThh:mm:ss where "CC" represents the century, "YY" the year, "MM" the month and "DD" the day, preceded by an optional leading "-" sign to indicate a negative number. If the sign is omitted, "+" is assumed. The letter "T" is the date/time separator and "hh", "mm", "ss" represent hour, minute and second respectively. Additional digits can be used to increase the precision of fractional seconds if desired i.e the format ss.ss... with any number of digits after the decimal point is supported.

```

</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:BaseTimeDataType"/>
  </complexContent>
</complexType>
<complexType name="BaseDataType" abstract="true">
  <annotation>
    <documentation>An abstract type used by within the schema to derive other data types by the ground
system. </documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:NameDescriptionType">
      <sequence>
        <element name="UnitSet">
          <complexType>
            <sequence>
              <element name="Unit" type="xtce:UnitType" minOccurs="0" maxOccurs="unbounded"/>
            </sequence>
          </complexType>
        </element>
        <choice minOccurs="0">
          <element name="BinaryDataEncoding" type="xtce:BinaryDataEncodingType"/>
          <element name="FloatDataEncoding" type="xtce:FloatDataEncodingType"/>
          <element name="IntegerDataEncoding" type="xtce:IntegerDataEncodingType"/>
          <element name="StringDataEncoding" type="xtce:StringDataEncodingType"/>
        </choice>
      </sequence>
      <attribute name="baseType" type="xtce:NameReferenceType">
        <annotation>
          <documentation>Used to derive one Data Type from another - will inherit all the attributes from
the baseType any of which may be redefined in this type definition. </documentation>
          <appinfo>Must be derived from a like type (e.g., Sting from String). No circular derivations.
        </appinfo>
      </attribute>
    </extension>
  </complexContent>
</complexType>
<complexType name="BaseTimeDataType" abstract="true">
  <annotation>
    <documentation>An abstract type used by within the schema to describe derive other data types by the
ground system. </documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:NameDescriptionType">
      <sequence>
        <sequence minOccurs="0">
          <element name="Encoding">
            <annotation>
              <documentation>Scale and offset are used in a y =mx +b type relationship (m is the
scale and b is the offset) to make adjustmets to the encoded value to that it matches the time units. Binary Encoded time
is typically used with a user supplied transform algorithm to convert time data formats that are too difficult to describe in
XTCE.</documentation>
            </annotation>
          </element>
        </sequence>
        <choice>
          <element name="BinaryDataEncoding" type="xtce:BinaryDataEncodingType"/>
          <element name="FloatDataEncoding" type="xtce:FloatDataEncodingType"/>
          <element name="IntegerDataEncoding" type="xtce:IntegerDataEncodingType"/>
          <element name="StringDataEncoding" type="xtce:StringDataEncodingType"/>
        </choice>
        <attribute name="units" type="xtce:TimeUnits" default="seconds"/>
        <attribute name="scale" type="double" default="1"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

```

        <attribute name="offset" type="double" default="0"/>
      </complexType>
    </element>
  </sequence>
  <sequence minOccurs="0">
    <element name="ReferenceTime" type="xtce:ReferenceTimeType"/>
  </sequence>
  </sequence>
  <attribute name="initialValue" type="duration"/>
</extension>
</complexContent>
</complexType>
<complexType name="BinaryDataType">
  <annotation>
    <documentation>Contains an arbitrarily large binary value </documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:BaseDataType">
      <attribute name="initialValue" type="hexBinary">
        <annotation>
          <documentation>Extra bits are truncated from the MSB (leftmost)</documentation>
        </annotation>
      </attribute>
    </extension>
  </complexContent>
</complexType>
<complexType name="EnumeratedDataType">
  <annotation>
    <documentation>Contains an enumerated value - a value that has both an integral and a string
representation.</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:BaseDataType">
      <sequence>
        <element name="EnumerationList">
          <complexType>
            <sequence>
              <element name="Enumeration" type="xtce:ValueEnumerationType"
maxOccurs="unbounded"/>
            </sequence>
          </complexType>
        </element>
      </sequence>
      <attribute name="initialValue" type="string">
        <annotation>
          <documentation>Initial value is always given in calibrated form.</documentation>
        </annotation>
      </attribute>
    </extension>
  </complexContent>
</complexType>
<complexType name="FloatDataType">
  <annotation>
    <documentation>Contains a floating point value</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:NumericDataType">
      <attribute name="initialValue" type="decimal">
        <annotation>
          <documentation>Initial value is always given in calibrated form</documentation>
        </annotation>
      </attribute>
      <attribute name="sizeInBits" default="32">
        <simpleType>
          <restriction base="positiveInteger">
            <enumeration value="32"/>
            <enumeration value="64"/>
            <enumeration value="128"/>
          </restriction>
        </simpleType>
      </attribute>
    </extension>
  </complexContent>
</complexType>

```

```

        </attribute>
      </extension>
    </complexContent>
  </complexType>
  <complexType name="BooleanDataType">
    <annotation>
      <documentation>Contains a boolean value</documentation>
    </annotation>
    <complexContent>
      <extension base="xtce:BaseDataType">
        <attribute name="initialValue" type="string">
          <annotation>
            <documentation>Initial value is always given in calibrated form. </documentation>
            <appinfo>Initial value must match either the oneStringValue or the zeroStringValue</appinfo>
          </annotation>
        </attribute>
        <attribute name="oneStringValue" type="string" default="True"/>
        <attribute name="zeroStringValue" type="string" default="False"/>
      </extension>
    </complexContent>
  </complexType>
  <complexType name="IntegerDataType">
    <annotation>
      <documentation>Contains an integral value</documentation>
    </annotation>
    <complexContent>
      <extension base="xtce:NumericDataType">
        <attribute name="initialValue" type="integer">
          <annotation>
            <documentation>Initial value is always given in calibrated form. Default is base 10 form; binary,
            octal, or hexadecimal values may be given by preceding value with 0b, 0o, 0x respectively.</documentation>
          </annotation>
        </attribute>
        <attribute name="sizeInBits" type="positiveInteger" default="32"/>
        <attribute name="signed" type="boolean" default="true"/>
      </extension>
    </complexContent>
  </complexType>
  <complexType name="NumericDataType">
    <annotation>
      <documentation>An abstract type that is a super type of either an Integer or Float Data
      type.</documentation>
    </annotation>
    <complexContent>
      <extension base="xtce:BaseDataType">
        <sequence>
          <element name="ToString" minOccurs="0">
            <complexType>
              <complexContent>
                <extension base="xtce:NumberToStringType"/>
              </complexContent>
            </complexType>
          </element>
          <element name="ValidRange" minOccurs="0">
            <annotation>
              <documentation>The Valid Range bounds the universe of possible values this Parameter
              may have.</documentation>
            </annotation>
            <complexType>
              <complexContent>
                <extension base="xtce:DecimalRangeType"/>
              </complexContent>
            </complexType>
          </element>
          <element name="DefaultCalibrator" type="xtce:CalibratorType" minOccurs="0"/>
          <element name="ContextCalibratorList" minOccurs="0">
            <complexType>
              <sequence>
                <element name="ContextCalibrator" type="xtce:ContextCalibratorType"
                maxOccurs="unbounded"/>
              </sequence>
            </complexType>
          </element>
        </sequence>
      </extension>
    </complexContent>
  </complexType>

```

```

        </sequence>
      </complexType>
    </element>
  </sequence>
  <attribute name="validRangeAppliesToCalibrated" type="boolean" default="true"/>
</extension>
</complexContent>
</complexType>
<complexType name="StringDataType">
  <annotation>
    <documentation>Contains a String Value</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:BaseDataType">
      <sequence>
        <element name="SizeRangeInCharacters" type="xtce:IntegerRangeType" minOccurs="0"/>
      </sequence>
      <attribute name="initialValue" type="string">
        <annotation>
          <documentation/>
        </annotation>
      </attribute>
      <attribute name="restrictionPattern" type="string">
        <annotation>
          <documentation>restriction pattern is a regular expression</documentation>
        </annotation>
      </attribute>
      <attribute name="characterWidth">
        <simpleType>
          <restriction base="integer">
            <enumeration value="8"/>
            <enumeration value="16"/>
          </restriction>
        </simpleType>
      </attribute>
    </extension>
  </complexContent>
</complexType>
<complexType name="DataEncodingType">
  <annotation>
    <documentation xml:lang="en">Describes how a particular piece of data is sent or received from some non-
native, off-platform device. (e.g. a spacecraft)</documentation>
  </annotation>
  <sequence>
    <element name="ErrorDetectCorrect" type="xtce:ErrorDetectCorrectType" minOccurs="0"/>
    <element name="ByteOrderList" type="xtce:ByteOrderType" minOccurs="0">
      <annotation>
        <documentation>Used to describe an arbitrary byte order in multibyte parameters. First byte in list is
the first in the stream. Byte significance is the is highest for most significant bytes. If not included, it is assumed that the
most significant byte is first, least significant byte last.</documentation>
      </annotation>
    </element>
  </sequence>
  <attribute name="bitOrder" default="mostSignificantBitFirst">
    <simpleType>
      <restriction base="string">
        <enumeration value="leastSignificantBitFirst"/>
        <enumeration value="mostSignificantBitFirst"/>
      </restriction>
    </simpleType>
  </attribute>
</complexType>
<complexType name="IntegerDataEncodingType">
  <annotation>
    <documentation xml:lang="en">For all major encodings of integer data</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:DataEncodingType">
      <sequence>
        <element name="DefaultCalibrator" type="xtce:CalibratorType" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

```

        <element name="ContextCalibratorList" minOccurs="0">
            <annotation>
                <documentation>Use when different calibrations must be used on the Parameter in different
contexts. Use the first one that tests true</documentation>
            </annotation>
            <complexType>
                <sequence>
                    <element name="ContextCalibrator" type="xtce:ContextCalibratorType"
maxOccurs="unbounded"/>
                </sequence>
            </complexType>
        </element>
    </sequence>
    <attribute name="encoding" default="unsigned">
        <simpleType>
            <restriction base="string">
                <enumeration value="unsigned"/>
                <enumeration value="signMagnitude"/>
                <enumeration value="twosCompliment"/>
                <enumeration value="onesCompliment"/>
                <enumeration value="BCD"/>
                <enumeration value="packedBCD"/>
            </restriction>
        </simpleType>
    </attribute>
    <attribute name="sizeInBits" type="positiveInteger" default="8"/>
</extension>
</complexContent>
</complexType>
<complexType name="FloatDataEncodingType">
    <annotation>
        <documentation xml:lang="en">For common encodings of floating point data</documentation>
    </annotation>
    <complexContent>
        <extension base="xtce:DataEncodingType">
            <sequence>
                <element name="DefaultCalibrator" type="xtce:CalibratorType" minOccurs="0"/>
                <element name="ContextCalibratorList" minOccurs="0">
                    <annotation>
                        <documentation>Use when different calibrations must be used on the Parameter in different
contexts. Use the first one that tests true</documentation>
                    </annotation>
                    <complexType>
                        <sequence>
                            <element name="ContextCalibrator" type="xtce:ContextCalibratorType"
maxOccurs="unbounded"/>
                        </sequence>
                    </complexType>
                </element>
            </sequence>
            <attribute name="encoding" default="IEEE754_1985">
                <simpleType>
                    <restriction base="string">
                        <enumeration value="IEEE754_1985"/>
                        <enumeration value="MILSTD_1750A"/>
                    </restriction>
                </simpleType>
            </attribute>
            <attribute name="sizeInBits" default="32">
                <simpleType>
                    <restriction base="positiveInteger">
                        <enumeration value="32"/>
                        <enumeration value="64"/>
                        <enumeration value="128"/>
                    </restriction>
                </simpleType>
            </attribute>
        </extension>
    </complexContent>
</complexType>

```

```

<complexType name="StringDataEncodingType">
  <annotation>
    <documentation xml:lang="en">For common encodings of string data</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:DataEncodingType">
      <sequence>
        <element name="DefaultCalibrator" type="xtce:CalibratorType" minOccurs="0"/>
        <element name="ContextCalibratorList" minOccurs="0">
          <annotation>
            <documentation>Use when different calibrations must be used on the Parameter in different
contexts. Use the first one that tests true</documentation>
          </annotation>
          <complexType>
            <sequence>
              <element name="ContextCalibrator" type="xtce:ContextCalibratorType"
maxOccurs="unbounded"/>
            </sequence>
          </complexType>
        </element>
        <element name="SizeInBits">
          <complexType>
            <choice>
              <element name="Fixed" type="xtce:IntegerValueType"/>
              <element name="TerminationChar" type="hexBinary">
                <annotation>
                  <documentation>Like C strings, they are terminated with a special string,
usually a null character.</documentation>
                </annotation>
                <!-- default="00" (does not work with CASTOR 0.9.5.3) -->
              </element>
              <element name="LeadingSize">
                <annotation>
                  <documentation>Like PASCAL strings, the size of the string is given as an
integer at the start of the string. SizeTag must be an unsigned Integer</documentation>
                </annotation>
                <complexType>
                  <attribute name="sizeInBitsOfSizeTag" type="positiveInteger" default="16"/>
                </complexType>
              </element>
            </choice>
          </complexType>
        </element>
      </sequence>
      <attribute name="encoding" default="UTF-8">
        <simpleType>
          <restriction base="string">
            <enumeration value="UTF-8"/>
            <enumeration value="UTF-16"/>
          </restriction>
        </simpleType>
      </attribute>
    </extension>
  </complexContent>
</complexType>
<complexType name="BinaryDataEncodingType">
  <annotation>
    <documentation xml:lang="en">For binary data or for integer, float, string, or time data that is not in any of
the known encoding formats. For any data that is not encoded in any of the known integer, float, string, or time data
formats use a To/From transform algorithm.</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:DataEncodingType">
      <sequence>
        <element name="SizeInBits" type="xtce:IntegerValueType"/>
        <element name="FromBinaryTransformAlgorithm" type="xtce:InputAlgorithmType" minOccurs="0">
          <annotation>
            <documentation>Used to convert binary data to an application data type</documentation>
          </annotation>
        </element>
      </sequence>
    </extension>
  </complexContent>

```

```

        <element name="ToBinaryTransformAlgorithm" type="xtce:InputAlgorithmType" minOccurs="0">
            <annotation>
                <documentation>Used to convert binary data from an application data type to binary
data</documentation>
            </annotation>
        </element>
    </sequence>
</extension>
</complexContent>
</complexType>
<simpleType name="EpochType">
    <annotation>
        <documentation>Epochs may be specified as a date or TAI (which correlates to 1 January
1958)</documentation>
    </annotation>
    <union memberTypes="date">
        <simpleType>
            <restriction base="string">
                <enumeration value="TAI"/>
            </restriction>
        </simpleType>
    </union>
</simpleType>
<!-- ***** DataTypes-->
<!-- ***** Common Types Schema -->
<!-- Basic elements used for in all dictionaries -->
<complexType name="AliasSetType">
    <annotation>
        <documentation>Contains an unordered collection of Alias's</documentation>
    </annotation>
    <sequence>
        <element name="Alias" minOccurs="0" maxOccurs="unbounded">
            <annotation>
                <documentation xml:lang="en">Used to contain an alias (alternate) name or ID for the the object.
For example, a parameter may have a mnemonic, an on-board id, and special IDs used by various ground software
applications; all of these are alias's. Some ground system processing equipment has some severe naming restrictions on
parameters (e.g., names must less then 12 characters, single case or integral id's only); their alias's provide a means of
capturing each name in a "nameSpace".</documentation>
            </annotation>
            <complexType>
                <attribute name="nameSpace" type="string" use="required"/>
                <attribute name="alias" type="string" use="required"/>
            </complexType>
        </element>
    </sequence>
</complexType>
<complexType name="ANDedConditionsType">
    <annotation>
        <documentation>A list of boolean comparisons, or boolean groups that are logically ANDed together. Any
ORed conditions in the list are evaluated first.</documentation>
    </annotation>
    <choice minOccurs="2" maxOccurs="unbounded">
        <element name="Condition" type="xtce:ComparisonCheckType"/>
        <element name="ORedConditions" type="xtce:ORedConditionsType"/>
    </choice>
</complexType>
<simpleType name="BinaryType">
    <annotation>
        <documentation>A simple restriction on string for hexadecimal numbers. Must be in 0b or 0B
form.</documentation>
    </annotation>
    <restriction base="string">
        <pattern value="0[bB][0-1]+"/>
    </restriction>
</simpleType>
<complexType name="BooleanExpressionType">
    <annotation>
        <documentation>Holds an arbitrarily complex boolean expression</documentation>
    </annotation>

```

```

<choice>
  <element name="Condition" type="xtce:ComparisonCheckType"/>
  <element name="ANDedConditions" type="xtce:ANDedConditionsType"/>
  <element name="ORedConditions" type="xtce:ORedConditionsType"/>
</choice>
</complexType>
<complexType name="ByteOrderType">
  <annotation>
    <documentation>An ordered list of bytes where the order of the bytes is in stream order. Each byte has an
attribute giving its significance.</documentation>
    <appinfo>The software must check to ensure that the significance of each byte is unique, and does not
contain bytes of greater significance greater than the size of the object</appinfo>
  </annotation>
  <sequence minOccurs="2" maxOccurs="unbounded">
    <element name="Byte">
      <complexType>
        <attribute name="byteSignificance" type="nonNegativeInteger" use="required"/>
      </complexType>
    </element>
  </sequence>
</complexType>
<complexType name="ComparisonCheckType">
  <annotation>
    <documentation xml:lang="en">A ParameterInstanceRef to a value or another parameter
instance</documentation>
  </annotation>
  <sequence>
    <element name="ParameterInstanceRef" type="xtce:ParameterInstanceRefType"/>
    <element name="ComparisonOperator" type="xtce:ComparisonOperatorsType"/>
    <choice>
      <element name="ParameterInstanceRef" type="xtce:ParameterInstanceRefType">
        <annotation>
          <documentation>Parameter is assumed to be of the same type as the comparison
Parameter</documentation>
        </annotation>
      </element>
      <element name="Value" type="string">
        <annotation>
          <documentation>Value is assumed to be of the same type as the comparison
Parameter</documentation>
        </annotation>
      </element>
    </choice>
  </sequence>
</complexType>
<complexType name="ComparisonType">
  <annotation>
    <documentation>A simple ParameterInstanceRef to value comparison. The string supplied in the value
attribute needs to be converted to a type matching the Parameter being compared to. Numerical values are assumed to
be base 10 unless preceded by 0x (hexadecimal), 0o (octal), or 0b (binary). The value is truncated to use the least
significant bits that match the bit size of the Parameter being compared to.</documentation>
  </annotation>
  <simpleContent>
    <extension base="xtce:ParameterInstanceRefType">
      <attribute name="comparisonOperator" type="xtce:ComparisonOperatorsType" default="="/>
      <attribute name="value" type="string" use="required"/>
    </extension>
  </simpleContent>
</complexType>
<simpleType name="ComparisonOperatorsType">
  <annotation>
    <documentation xml:lang="en">Operators to use when testing a boolean condition for a validity
check</documentation>
  </annotation>
  <restriction base="string">
    <enumeration value="="/>
    <enumeration value="!=">
    <enumeration value="&lt;"/>
    <enumeration value="&lt;=">
    <enumeration value=">"/>
  </restriction>

```

```

        <enumeration value=">="/>
    </restriction>
</simpleType>
<complexType name="ContextCalibratorType">
    <annotation>
        <documentation>Context calibrations are applied when the ContextMatch is true. Context calibrators override
Default calibrators</documentation>
    </annotation>
    <sequence>
        <element name="ContextMatch" type="xtce:MatchCriteriaType"/>
        <element name="Calibrator" type="xtce:CalibratorType"/>
        <!-- <element name="Context" type="xtce:MatchCriteriaType"/> -->
    </sequence>
</complexType>
<complexType name="DecimalValueType">
    <annotation>
        <documentation>Contains an Numeric value; value may be provided directly or via the value in a
parameter.</documentation>
    </annotation>
    <choice>
        <element name="FixedValue" type="decimal"/>
        <element name="DynamicValue">
            <annotation>
                <documentation>Uses a parameter to for the value. The parameter value may be optionally
adjusted by a Linear function or use a series of boolean expressions to lookup the value. Anything more complex and a
DynamicValue with a CustomAlgorithm may be used </documentation>
            </annotation>
            <complexType>
                <sequence>
                    <element name="ParameterInstanceRef" type="xtce:ParameterInstanceRefType"/>
                    <element name="LinearAdjustment" minOccurs="0">
                        <annotation>
                            <documentation>A slope and intercept may be applied to scale or shift the value of the
parameter in the dynamic value</documentation>
                        </annotation>
                        <complexType>
                            <attribute name="slope" type="decimal" default="0"/>
                            <attribute name="intercept" type="decimal" default="0"/>
                        </complexType>
                    </element>
                </sequence>
            </complexType>
        </element>
    </choice>
</complexType>
<complexType name="DescriptionType" abstract="true">
    <annotation>
        <documentation xml:lang="en">An abstract type definition used as the base for NameDescriptionType or
OptionalNameDescriptionType. The short description is intended to be used for quick "memory jogger" descriptions of the
object. </documentation>
    </annotation>
    <sequence>
        <element name="LongDescription" type="string" minOccurs="0">
            <annotation>
                <documentation>The Long Description is intended to be used for explanatory descriptions of the
object and may include HTML markup. Long Descriptions are of unbounded length</documentation>
            </annotation>
        </element>
        <element name="AliasSet" type="xtce:AliasSetType" minOccurs="0"/>
        <element name="AncillaryDataSet" minOccurs="0">
            <complexType>
                <sequence>
                    <element name="AncillaryData" maxOccurs="unbounded">
                        <annotation>
                            <documentation>Use for any other data associated with each named object. May be
used to include administrative data (e.g., version, CM or tags) or potentially any MIME type. Data may be included or
given as an href. </documentation>
                        </annotation>
                    </complexType>
                </sequence>
            </complexType>
        </element>
    </sequence>

```

```

        <extension base="string">
            <attribute name="name" type="string" use="required"/>
            <attribute name="mimeType" type="string" default="text/plain"/>
            <attribute name="href" type="anyURI"/>
        </extension>
    </simpleContent>
</complexType>
</element>
</sequence>
</complexType>
</element>
</sequence>
<attribute name="shortDescription" type="string" use="optional">
    <annotation>
        <documentation>It is strongly recommended that the short description be kept under 80 characters in
length</documentation>
    </annotation>
</attribute>
</complexType>
<complexType name="ErrorDetectCorrectType">
    <annotation>
        <documentation xml:lang="en">A simple element that provides for simple, but common error checking and
detection.</documentation>
    </annotation>
    <choice>
        <element name="Parity">
            <annotation>
                <documentation xml:lang="en">Bit position starts with 'zero'.</documentation>
            </annotation>
            <complexType>
                <attribute name="type" use="required">
                    <simpleType>
                        <restriction base="string">
                            <enumeration value="Even"/>
                            <enumeration value="Odd"/>
                        </restriction>
                    </simpleType>
                </attribute>
                <attribute name="bitsFromReference" type="nonNegativeInteger" use="required"/>
                <attribute name="reference" default="start">
                    <simpleType>
                        <restriction base="string">
                            <enumeration value="start"/>
                            <enumeration value="end"/>
                        </restriction>
                    </simpleType>
                </attribute>
            </complexType>
        </element>
        <element name="CRC">
            <annotation>
                <documentation xml:lang="en">Cyclic Redundancy Check definition. Legal values for coefficient's
are 0 or 1. Exponents must be integer values.</documentation>
            </annotation>
            <complexType>
                <sequence>
                    <element name="Polynomial" type="xtce:PolynomialType"/>
                </sequence>
                <attribute name="bitsFromReference" type="nonNegativeInteger"/>
                <attribute name="reference" default="start">
                    <simpleType>
                        <restriction base="string">
                            <enumeration value="start"/>
                            <enumeration value="end"/>
                        </restriction>
                    </simpleType>
                </attribute>
            </complexType>
        </element>
    </choice>

```

```

</complexType>
<simpleType name="FixedIntegerValueType">
  <annotation>
    <documentation>A simple union type combining integer, octal, binary, and hexadecimal
types</documentation>
  </annotation>
  <union memberTypes="integer xtce:HexadecimalType xtce:OctalType xtce:BinaryType"/>
</simpleType>
<complexType name="HeaderType">
  <annotation>
    <documentation xml:lang="en">Schema for a Header record. A header contains general information about
the system or subsystem.</documentation>
  </annotation>
  <sequence>
    <element name="AuthorSet" minOccurs="0">
      <complexType>
        <sequence>
          <element name="Author" type="string" minOccurs="0" maxOccurs="unbounded"/>
        </sequence>
      </complexType>
    </element>
    <element name="NoteSet" minOccurs="0">
      <complexType>
        <sequence>
          <element name="Note" type="string" minOccurs="0" maxOccurs="unbounded"/>
        </sequence>
      </complexType>
    </element>
    <element name="HistorySet" minOccurs="0">
      <complexType>
        <sequence>
          <element name="History" type="string" minOccurs="0" maxOccurs="unbounded"/>
        </sequence>
      </complexType>
    </element>
  </sequence>
  <attribute name="version" type="string"/>
  <attribute name="date" type="string"/>
  <attribute name="classification" type="string" default="NotClassified"/>
  <attribute name="classificationInstructions" type="string"/>
  <attribute name="validationStatus" use="required">
    <simpleType>
      <restriction base="string">
        <enumeration value="Unknown"/>
        <enumeration value="Working"/>
        <enumeration value="Draft"/>
        <enumeration value="Test"/>
        <enumeration value="Validated"/>
        <enumeration value="Released"/>
        <enumeration value="Withdrawn"/>
      </restriction>
    </simpleType>
  </attribute>
</complexType>
<simpleType name="HexadecimalType">
  <annotation>
    <documentation>A simple restriction on string for hexadecimal numbers. Must be in 0x or 0X
form.</documentation>
  </annotation>
  <restriction base="string">
    <pattern value="0[xX][0-9a-fA-F]+"/>
  </restriction>
</simpleType>
<complexType name="IntegerValueType">
  <annotation>
    <documentation>Contains an Integer value; value may be provided directly or via the value in a
parameter.</documentation>
  </annotation>
  <choice>
    <element name="FixedValue" type="xtce:FixedIntegerValueType"/>

```

```

<element name="DynamicValue">
  <annotation>
    <documentation>Uses a parameter to for the value. The parameter value may be optionally
adjusted by a Linear function or use a series of boolean expressions to lookup the value. Anything more complex and a
DynamicValue with a CustomAlgorithm may be used </documentation>
  </annotation>
  <complexType>
    <sequence>
      <element name="ParameterInstanceRef" type="xtce:ParameterInstanceRefType"/>
      <element name="LinearAdjustment" minOccurs="0">
        <annotation>
          <documentation>A slope and intercept may be applied to scale or shift the value of the
parameter in the dynamic value</documentation>
        </annotation>
        <complexType>
          <attribute name="slope" type="integer" default="0"/>
          <attribute name="intercept" type="integer" default="0"/>
        </complexType>
      </element>
    </sequence>
  </complexType>
</element>
<element name="DiscreteLookupList">
  <annotation>
    <documentation>Lookup a value using the lookup list supplied. Use the first match
found.</documentation>
  </annotation>
  <complexType>
    <sequence>
      <element name="DiscreteLookup" maxOccurs="unbounded">
        <complexType>
          <complexContent>
            <extension base="xtce:MatchCriteriaType">
              <attribute name="value" type="integer" use="required"/>
            </extension>
          </complexContent>
        </complexType>
      </element>
    </sequence>
  </complexType>
</element>
</choice>
</complexType>
<simpleType name="MathOperatorsType">
  <annotation>
    <documentation xml:lang="en">Mathematical operators</documentation>
  </annotation>
  <restriction base="string">
    <enumeration value="+"/>
    <enumeration value="-"/>
    <enumeration value="mult"/>
    <enumeration value="div"/>
    <enumeration value="mod"/>
    <enumeration value="exp"/>
    <enumeration value="bitor"/>
    <enumeration value="bitand"/>
    <enumeration value="bitxor"/>
  </restriction>
</simpleType>
<complexType name="MatchCriteriaType">
  <annotation>
    <documentation>Contains either a simple Comparison, a ComparisonList, an arbitrarily complex
BooleanExpression or an escape to an externally defined algorithm</documentation>
  </annotation>
  <choice>
    <element name="Comparison" type="xtce:ComparisonType">
      <annotation>
        <documentation>A simple comparison check</documentation>
      </annotation>
    </element>

```

```

<element name="ComparisonList">
  <annotation>
    <documentation>All comparisons must be true</documentation>
  </annotation>
  <complexType>
    <sequence>
      <element name="Comparison" type="xtce:ComparisonType" maxOccurs="unbounded"/>
    </sequence>
  </complexType>
</element>
<element name="BooleanExpression" type="xtce:BooleanExpressionType">
  <annotation>
    <documentation>An arbitrarily complex boolean expression</documentation>
  </annotation>
</element>
<element name="CustomAlgorithm" type="xtce:InputAlgorithmType">
  <annotation>
    <documentation>An escape to an externally defined algorithm</documentation>
  </annotation>
</element>
</choice>
</complexType>
<complexType name="MathOperationType">
  <annotation>
    <documentation xml:lang="en">A simple math operation</documentation>
  </annotation>
  <sequence>
    <choice>
      <element name="ParameterInstanceRef" type="xtce:ParameterInstanceRefType"/>
      <element name="Value" type="string">
        <annotation>
          <documentation>Value is assumed to be of the same type as the Parameter</documentation>
        </annotation>
      </element>
    </choice>
    <sequence minOccurs="0">
      <element name="Operator" type="xtce:MathOperatorsType">
        <choice>
          <element name="ParameterInstanceRef" type="xtce:ParameterInstanceRefType"/>
          <element name="Value" type="string">
            <annotation>
              <documentation>Value is assumed to be of the same type as the
Parameter</documentation>
            </annotation>
          </element>
        </choice>
      </sequence>
    </sequence>
  </complexType>
<simpleType name="NameType">
  <annotation>
    <documentation xml:lang="en">Used for "directory" style unique names. We need to preclude '.', '/', ':', '"', '['
and ']'. Only letters, digits, '_', '' and "-" are allowed </documentation>
  </annotation>
  <restriction base="string">
    <pattern value="[a-zA-Z0-9_\- ]*"/>
  </restriction>
</simpleType>
<complexType name="NameDescriptionType">
  <annotation>
    <documentation xml:lang="en">The type definition used by most elements that require a name with optional
descriptions. </documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:DescriptionType">
      <attribute name="name" type="xtce:NameType" use="required"/>
    </extension>
  </complexContent>
</complexType>
<simpleType name="NameReferenceType">

```

```

        <annotation>
        <documentation xml:lang="en">Used when referencing a directory style "NameType". All characters are
        legal. All name references use a Unix 'like' name referencing mechanism across the SpaceSystem Tree (e.g.,
        SimpleSat/Bus/EPDS/BatteryOne/Voltage) where the '/', '.', and '.' are used to navigate through the hierarchy. The use of
        an unqualified name will search for an item in the current SpaceSystem first, then if none is found, in progressively higher
        SpaceSystems. A SpaceSystem is a name space (i.e., a named type declared in MetaCommandData is also declared in
        TelemetryMetaData - and vice versa).</documentation>
        </annotation>
        <restriction base="string"/>
    </simpleType>
    <complexType name="NumberToStringType">
        <annotation>
        <documentation xml:lang="en">There are two ways numeric data can be changed to string data: using a
        Java style NumberFormat, or using an enumerated list. Enumerated lists can be assigned to a single value or a value
        range.</documentation>
        </annotation>
        <complexContent>
        <extension base="xtce:OptionalNameDescriptionType">
            <choice>
                <choice maxOccurs="unbounded">
                    <element name="ValueEnumeration" type="xtce:ValueEnumerationType">
                        <annotation>
                        <documentation xml:lang="en">A number or range assigned to a
        string.</documentation>
                        </annotation>
                    </element>
                    <element name="RangeEnumeration">
                        <annotation>
                        <documentation xml:lang="en">A string value associated with a numerical
        range.</documentation>
                        </annotation>
                    </element>
                </choice>
            </extension>
        </complexContent>
        </complexType>
        <complexType>
        <complexContent>
        <extension base="xtce:DecimalRangeType">
            <attribute name="label" type="string" use="required"/>
        </extension>
        </complexContent>
        </complexType>
        </element>
    </choice>
    <element name="NumberFormat">
        <complexType>
        <attribute name="numberBase" type="xtce:RadixType" use="optional"/>
        <attribute name="minimumFractionDigits" type="nonNegativeInteger" use="optional"/>
        <attribute name="maximumFractionDigits" type="nonNegativeInteger" use="optional"/>
        <attribute name="minimumIntegerDigits" type="nonNegativeInteger" use="optional"/>
        <attribute name="maximumIntegerDigits" type="nonNegativeInteger" use="optional"/>
        <attribute name="negativeSuffix" type="string" use="optional"/>
        <attribute name="positiveSuffix" type="string" use="optional"/>
        <attribute name="negativePrefix" type="string" use="optional" default="-"/>
        <attribute name="positivePrefix" type="string" use="optional"/>
        <attribute name="showThousandsGrouping" type="boolean" use="optional" default="true"/>
        <attribute name="notation" use="optional" default="normal">
            <simpleType>
            <restriction base="string">
                <enumeration value="normal"/>
                <enumeration value="scientific"/>
                <enumeration value="engineering"/>
            </restriction>
            </simpleType>
        </attribute>
        </complexType>
    </element>
</choice>
</extension>
</complexContent>
</complexType>
<simpleType name="OctalType">
    <annotation>

```

```

        <documentation>A simple restriction on string for hexadecimal numbers. Must be in 0o or 0O
form.</documentation>
    </annotation>
    <restriction base="string">
        <pattern value="0[oO][0-7]+"/>
    </restriction>
</simpleType>
<complexType name="OptionalNameDescriptionType">
    <annotation>
        <documentation xml:lang="en">The type definition used by most elements that have an optional name with
optional descriptions. </documentation>
    </annotation>
    <complexContent>
        <extension base="xtce:DescriptionType">
            <attribute name="name" type="xtce:NameType" use="optional"/>
        </extension>
    </complexContent>
</complexType>
<complexType name="ORedConditionsType">
    <annotation>
        <documentation>A list of boolean comparisons, or boolean groups that are logically ORed together. Any
ANDed conditions in the list are evaluated first.</documentation>
    </annotation>
    <choice minOccurs="2" maxOccurs="unbounded">
        <element name="Condition" type="xtce:ComparisonCheckType"/>
        <element name="ANDedConditions" type="xtce:ANDedConditionsType"/>
    </choice>
</complexType>
<complexType name="ParameterSetType">
    <annotation>
        <documentation>Used by both the TelemetryMetaData and the CommandMetaData components each may
be built independently.</documentation>
    </annotation>
    <choice maxOccurs="unbounded">
        <element name="Parameter">
            <annotation>
                <appinfo>Need to ensure that the named types actually exist</appinfo>
            </annotation>
            <complexType>
                <complexContent>
                    <extension base="xtce:NameDescriptionType">
                        <sequence>
                            <element name="ParameterProperties" type="xtce:ParameterPropertiesType"
minOccurs="0"/>
                        </sequence>
                        <attribute name="parameterTypeRef" type="xtce:NameReferenceType" use="required"/>
                    </extension>
                </complexContent>
            </complexType>
        </element>
        <element name="ParameterRef" type="xtce:ParameterRefType">
            <annotation>
                <documentation>Used to include a Parameter defined in another sub-system in this sub-
system.</documentation>
            </annotation>
        </element>
    </choice>
</complexType>
<complexType name="PolynomialType">
    <annotation>
        <documentation xml:lang="en">A polynomial expression. For example: 3 + 2x</documentation>
    </annotation>
    <sequence>
        <element name="Term" maxOccurs="unbounded">
            <annotation>
                <documentation xml:lang="en">A term in a polynomial expresssion. </documentation>
            </annotation>
            <complexType>
                <attribute name="coefficient" type="double" use="required"/>
                <attribute name="exponent" type="double" use="required"/>
            </complexType>
        </element>
    </sequence>

```

```

        </complexType>
      </element>
    </sequence>
  </complexType>
  <simpleType name="RadixType">
    <annotation>
      <documentation xml:lang="en">Specifies the number base</documentation>
    </annotation>
    <restriction base="string">
      <enumeration value="Decimal"/>
      <enumeration value="Hexadecimal"/>
      <enumeration value="Octal"/>
      <enumeration value="Binary"/>
    </restriction>
  </simpleType>
  <complexType name="ReferenceTimeType">
    <annotation>
      <documentation>Most time values are relative to another time e.g. seconds are relative to minutes, minutes
      are relative to hours. This type is used to describe this relationship starting with the least significant time Parameter to
      and progressing to the most significant time parameter. </documentation>
    </annotation>
    <choice>
      <element name="OffsetFrom" type="xtce:ParameterInstanceRefType"/>
      <element name="Epoch" type="xtce:EpochType"/>
    </choice>
  </complexType>
  <simpleType name="RelativeTimeType">
    <annotation>
      <documentation>Used to describe a relative time. Normally used for time offsets. A Relative time is
      expressed as PnYn MnDTnH nMnS, where nY represents the number of years, nM the number of months, nD the number
      of days, 'T' is the date/time separator, nH the number of hours, nM the number of minutes and nS the number of seconds.
      The number of seconds can include decimal digits to arbitrary precision. For example, to indicate a duration of 1 year, 2
      months, 3 days, 10 hours, and 30 minutes, one would write: P1Y2M3DT10H30M. One could also indicate a duration of
      minus 120 days as: -P120D. An extension of Schema duration type. </documentation>
    </annotation>
    <restriction base="duration"/>
  </simpleType>
  <complexType name="RepeatType">
    <annotation>
      <documentation xml:lang="en">Hold a structure that can be repeated X times, where X is the
      Count</documentation>
    </annotation>
    <sequence>
      <element name="Count" type="xtce:IntegerValueType">
        <annotation>
          <documentation xml:lang="en">Value (either fixed or dynamic) that contains the count of repeated
          structures.</documentation>
        </annotation>
      </element>
      <element name="Offset" minOccurs="0">
        <annotation>
          <documentation xml:lang="en">Indicates the distance between repeating entries (the last bit of one
          entry to the start bit of the next entry)</documentation>
        </annotation>
        <complexType>
          <complexContent>
            <extension base="xtce:IntegerValueType">
              <attribute name="offsetSizeInBits" type="positiveInteger" default="1"/>
            </extension>
          </complexContent>
        </complexType>
      </element>
    </sequence>
  </complexType>
  <complexType name="SplinePointType">
    <annotation>
      <documentation xml:lang="en">a spline is a set on points from which a curve may be drawn to interpolate
      raw to calibrated values</documentation>
    </annotation>
    <attribute name="order" type="positiveInteger" default="1"/>

```

```

    <attribute name="raw" type="double" use="required"/>
    <attribute name="calibrated" type="double" use="required"/>
</complexType>
<simpleType name="TimeUnits">
  <annotation>
    <documentation>base time units. days, months, years have obvious ambiguity and should be
avoided</documentation>
  </annotation>
  <restriction base="string">
    <enumeration value="seconds"/>
    <enumeration value="picoSeconds"/>
    <enumeration value="days"/>
    <enumeration value="months"/>
    <enumeration value="years"/>
  </restriction>
</simpleType>
<complexType name="UnitType" mixed="true">
  <annotation>
    <documentation>Used to hold the unit(s) plus possibly the exponents for the units</documentation>
  </annotation>
  <attribute name="power" type="decimal" use="optional" default="1"/>
  <attribute name="factor" type="string" default="1"/>
  <attribute name="description" type="string"/>
</complexType>
<complexType name="ValueEnumerationType">
  <annotation>
    <documentation>Contains a value and an associated string label</documentation>
  </annotation>
  <attribute name="value" type="integer" use="required"/>
  <attribute name="label" type="string" use="required"/>
</complexType>
<!--Types used with alarms-->
<simpleType name="AlarmLevels">
  <annotation>
    <documentation>An enumerated list of the possible alarm levels</documentation>
  </annotation>
  <restriction base="string">
    <enumeration value="normal"/>
    <enumeration value="watch"/>
    <enumeration value="warning"/>
    <enumeration value="distress"/>
    <enumeration value="critical"/>
    <enumeration value="severe"/>
  </restriction>
</simpleType>
<complexType name="AlarmConditionsType">
  <annotation>
    <documentation>When the alarm is determined by boolean logic</documentation>
  </annotation>
  <sequence>
    <element name="WatchAlarm" type="xtce:MatchCriteriaType" minOccurs="0"/>
    <element name="WarningAlarm" type="xtce:MatchCriteriaType" minOccurs="0"/>
    <element name="DistressAlarm" type="xtce:MatchCriteriaType" minOccurs="0"/>
    <element name="CriticalAlarm" type="xtce:MatchCriteriaType" minOccurs="0"/>
    <element name="SevereAlarm" type="xtce:MatchCriteriaType" minOccurs="0"/>
  </sequence>
</complexType>
<complexType name="AlarmRangesType">
  <annotation>
    <documentation>Contains five ranges: Watch, Warning, Distress, Critical, and Severe each in increasing
severity. Normally, only the Warning and Critical ranges are used and the color yellow is associated with Warning and the
color red is associated with Critical. The ranges given are valid for numbers lower than the min and higher than the max
values. These ranges should not overlap, but if they do, assume the most severe range is to be applied. All ranges are
optional and it is quite allowed for there to be only one end of the range. </documentation>
  </annotation>
  <sequence>
    <element name="WatchRange" type="xtce:DecimalRangeType" minOccurs="0"/>
    <element name="WarningRange" type="xtce:DecimalRangeType" minOccurs="0"/>
    <element name="DistressRange" type="xtce:DecimalRangeType" minOccurs="0"/>
    <element name="CriticalRange" type="xtce:DecimalRangeType" minOccurs="0"/>
  </sequence>

```

```

        <element name="SevereRange" type="xtce:DecimalRangeType" minOccurs="0"/>
    </sequence>
    <attribute name="appliesToCalibratedValues" type="boolean" default="true"/>
</complexType>
<complexType name="AlarmType" abstract="true">
    <annotation>
        <documentation>Alarms associated with numeric data types</documentation>
    </annotation>
    <choice minOccurs="0">
        <element name="AlarmConditions" type="xtce:AlarmConditionsType">
            <annotation>
                <documentation>A MatchCriteria may be specified for each of the 5 alarm levels. Each level is
optional and the alarm should be the highest level to test true.</documentation>
            </annotation>
        </element>
        <element name="CustomAlarm" type="xtce:InputAlgorithmType">
            <annotation>
                <documentation>An escape for ridiculously complex alarm conditions. Will trigger on changes to the
containing Parameter. </documentation>
            </annotation>
        </element>
    </choice>
    <attribute name="minViolations" type="positiveInteger" default="1">
        <annotation>
            <documentation>Number of successive instances that meet the alarm conditions for the Alarm to
trigger.</documentation>
        </annotation>
    </attribute>
</complexType>
<complexType name="BinaryAlarmConditionType">
    <annotation>
        <documentation>Alarm conditions for Binary types</documentation>
    </annotation>
    <complexContent>
        <extension base="xtce:AlarmType"/>
    </complexContent>
</complexType>
<complexType name="BooleanAlarmType">
    <annotation>
        <documentation>Alarm conditions for Boolean types</documentation>
    </annotation>
    <complexContent>
        <extension base="xtce:AlarmType"/>
    </complexContent>
</complexType>
<complexType name="DecimalRangeType">
    <annotation>
        <documentation xml:lang="en">A range of numbers. "minInclusive", "minExclusive", "maxInclusive" and
"maxExclusive" attributes are borrowed from the W3C schema language.</documentation>
    </annotation>
    <attribute name="minInclusive" type="decimal"/>
    <attribute name="minExclusive" type="decimal"/>
    <attribute name="maxInclusive" type="decimal"/>
    <attribute name="maxExclusive" type="decimal"/>
</complexType>
<complexType name="EnumerationAlarmType">
    <annotation>
        <documentation>Alarm conditions for Enumerations</documentation>
        <appinfo source="An additional check needs to be performed to ensure that the enumeration values in the
alarms are legal enumeration values for the Parameter"/>
    </annotation>
    <complexContent>
        <extension base="xtce:AlarmType">
            <sequence>
                <element name="EnumerationAlarmList">
                    <complexType>
                        <sequence>
                            <element name="EnumerationAlarm" maxOccurs="unbounded">
                                <complexType>
                                    <attribute name="alarmLevel" type="xtce:AlarmLevels" use="required"/>
                                </complexType>
                            </element>
                        </sequence>
                    </complexType>
                </element>
            </sequence>
        </extension>
    </complexContent>
</complexType>

```

```

        <attribute name="enumerationValue" type="string" use="required"/>
    </complexType>
</element>
</sequence>
</complexType>
</element>
</sequence>
<attribute name="defaultAlarmLevel" type="xtce:AlarmLevels" default="normal"/>
</extension>
</complexContent>
</complexType>
<complexType name="IntegerRangeType">
    <annotation>
        <documentation xml:lang="en">An integral range of numbers. "min", and "max".</documentation>
    </annotation>
    <attribute name="min" type="integer"/>
    <attribute name="max" type="integer"/>
</complexType>
<complexType name="NumericContextAlarmType">
    <annotation>
        <documentation>Context alarms are applied when the ContextMatch is true. Context alarms override Default
alarms</documentation>
    </annotation>
    <complexContent>
        <extension base="xtce:NumericAlarmType">
            <sequence>
                <element name="ContextMatch" type="xtce:MatchCriteriaType"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>
<complexType name="NumericAlarmType">
    <annotation>
        <documentation>Alarms associated with numeric data types</documentation>
    </annotation>
    <complexContent>
        <extension base="xtce:AlarmType">
            <sequence>
                <element name="StaticAlarmRanges" type="xtce:AlarmRangesType" minOccurs="0">
                    <annotation>
                        <documentation>StaticAlarmRanges are used to trigger alarms when the parameter value
passes some threshold value</documentation>
                    </annotation>
                </element>
                <element name="ChangePerSecondAlarmRanges" type="xtce:AlarmRangesType" minOccurs="0">
                    <annotation>
                        <documentation>ChangePerSecondAlarmRanges are used to trigger alarms when the
parameter value's rate-of-change passes some threshold value. An alarm condition that triggers when the value changes
too fast (or too slow)</documentation>
                    </annotation>
                </element>
            </sequence>
        </extension>
    </complexContent>
</complexType>
<complexType name="StringAlarmType">
    <annotation>
        <documentation>Alarm conditions for Strings</documentation>
    </annotation>
    <complexContent>
        <extension base="xtce:AlarmType">
            <sequence>
                <element name="StringAlarmList">
                    <complexType>
                        <sequence>
                            <element name="StringAlarm" minOccurs="unbounded">
                                <annotation>
                                    <documentation>Pattern may be a regular expression</documentation>
                                </annotation>
                            </complexType>
                        </sequence>
                    </complexType>
                </element>
            </sequence>
        </extension>
    </complexContent>
</complexType>

```

```

        <attribute name="alarmLevel" type="xtce:AlarmLevels" use="required"/>
        <attribute name="matchPattern" type="string" use="required"/>
    </complexType>
</element>
</sequence>
</complexType>
</element>
</sequence>
<attribute name="defaultAlarmLevel" type="xtce:AlarmLevels" default="normal"/>
</extension>
</complexContent>
</complexType>
<complexType name="TimeAlarmType">
    <annotation>
        <documentation>Alarms associated with time data types</documentation>
    </annotation>
    <complexContent>
        <extension base="xtce:AlarmType">
            <sequence>
                <element name="StaticAlarmRanges" minOccurs="0">
                    <annotation>
                        <documentation>StaticAlarmRanges are used to trigger alarms when the parameter value
passes some threshold value</documentation>
                    </annotation>
                    <complexType>
                        <complexContent>
                            <extension base="xtce:AlarmRangesType">
                                <attribute name="timeUnits" type="xtce:TimeUnits" default="seconds"/>
                            </extension>
                        </complexContent>
                    </complexType>
                </element>
                <element name="ChangePerSecondAlarmRanges" minOccurs="0">
                    <annotation>
                        <documentation>ChangePerSecondAlarmRanges are used to trigger alarms when the
parameter value's rate-of-change passes some threshold value. An alarm condition that triggers when the value changes
too fast (or too slow)</documentation>
                    </annotation>
                    <complexType>
                        <complexContent>
                            <extension base="xtce:AlarmRangesType">
                                <attribute name="timeUnits" type="xtce:TimeUnits" default="seconds"/>
                            </extension>
                        </complexContent>
                    </complexType>
                </element>
            </sequence>
        </extension>
    </complexContent>
</complexType>
<complexType name="TimeAlarmConditionType">
    <annotation>
        <documentation>Alarm conditions for Time types</documentation>
    </annotation>
    <complexContent>
        <extension base="xtce:AlarmType"/>
    </complexContent>
</complexType>
<complexType name="TimeContextAlarmType">
    <annotation>
        <documentation>Context alarms are applied when the ContextMatch is true. Context alarms override Default
alarms</documentation>
    </annotation>
    <complexContent>
        <extension base="xtce:TimeAlarmType">
            <sequence>
                <element name="ContextMatch" type="xtce:MatchCriteriaType"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>

```

```
</complexType>  
<!--***** End of Common Types Schema -->  
</schema>
```

Annex B - Schema Style Notes

A number of conventions were developed and adopted during the authorship of the schema to make understanding the schema easier and the wording a presentation more consistent.

- Element and Type names begin with a capital letter.
- Type names end with the word "Type".
- Attribute names begin with a lowercase letter.
- Usually, when the UML class diagram references classes, W3C Elements are used, and whenever the UML references simple types (strings, ints), W3C Attributes are used. In general, attributes are preferred over elements because they're easier to deal with in SAX and DOM, but whenever the Element/Attribute may one day carry metadata, elements should be used. One exception is enumerated classes, because enumerations may be defined for attributes but not for elements.
- Bias toward self-describing names over short, bandwidth conserving ones.
- Use mixed case in names rather than underscores to combine multiple words (camelCase).
- A documentation annotation is included in every element and type definition. Annotations for a type are included with the type definition; use of the type is annotated in the element definition.
- Hints on units (for values with units) are provided in the names of attributes and elements (e.g. "dataRateInBPS" is preferred over "dataRate" OR "frameLengthInBits" is preferred over "frameLength").
- Major elements or any elements used multiple times are first defined with a complexType definition
- All collections are put inside either a "List" element or a "Set" Element depending on whether the collection is ordered or unordered.
- Simplicity in the XML files is favored over simplicity in the Schema
- Whenever an additional validity check must be performed that is not describable in the schema language, an appinfo annotation describes that validity check

Bibliography

- [1] *CCSDS Packet Telemetry*, CCSDS 102.0-B-4
- [2] *CCSDS Telecommand*, CCSDS 203.0-B-1
- [3] *Telemetry and Telecommand Packet Utilisation*, Draft 5.3, 5 Apr 2001, ECSS-E-70-41
- [4] *SCOS-2000 Database Import ICD*, Issue 5.0, 19 Jun 2001, S2K-MCS-ICD-0001-TOS-GCI
- [5] *Telemetric and Command Data Specification*, Space RFP-1, 20 Aug 2001, Space/01-04-01
- [6] *Packet Utilisation Standard*, Issue 1, May 1994, ESA PSS-07-101
- [7] *CCSDS Time Code Formats*, Issue 2, April 1990, CCSDS 301.0-B-2
- [8] *SCOS-2000 Synthetic Parameters Software User Manual*, Issue 3.1, September 2001, S2K-MCS-SUM-0019-TOS-GCI
- [9] *Telemetry Attributes Transfer Standard (TMATS)*, IRIG-106